

高等学校信息安全专业规划教材

计算机系统安全教程

曹天杰 李琳 黄石 编著

清华大学出版社

21 世纪高等学校信息安全专业规划教材

计算机系统安全教程

曹天杰　李　琳　黄　石　编著

清华大学出版社
北　京

内 容 简 介

本书全面而又系统地讲述了计算机系统安全的基本知识。本书注重理论与实践相结合，条理清晰。围绕认证、访问控制、机密性、完整性、可用性、不可否认性、安全审计与报警等安全服务的实现，介绍了系统安全、网络安全、应用安全各个层次计算机系统可能面临的威胁与防范措施，从纵深防御的角度分析了信息保障中的保护、检测、响应和恢复等方面的知识。

本书主要内容包括：绪论、实现安全服务、操作系统安全、数据库安全、漏洞检测、攻击与防范、防火墙、系统入侵检测与防御、电子邮件安全、IP 安全、Web 安全。

本书可以作为信息安全专业、信息对抗专业、计算机科学与技术专业、网络工程专业或其他相关专业的本科生和研究生教材，也可以作为信息系统安全领域的从业人员参考书。

图书在版编目(CIP)数据

计算机系统安全教程/曹天杰，李琳，黄石编著. —北京：清华大学出版社，2010.11
(21 世纪高等学校信息安全专业规划教材)
ISBN 978-7-302-22618-5

Ⅰ. ①计… Ⅱ. ①曹… ②李… ③黄… Ⅲ. ①电子计算机－安全技术－高等学校－教材
Ⅳ. ①TP309

中国版本图书馆 CIP 数据核字(2010)第 081861 号

责任编辑：魏江江　李玮琪
责任校对：时翠兰
责任印制：

出版发行：清华大学出版社　　**地　　址**：北京清华大学学研大厦 A 座
http://www.tup.com.cn　　**邮　　编**：100084
社　总　机：010-62770175　　**邮　　购**：010-62786544
投稿与读者服务：010-62795954，jsjjc@tup.tsinghua.edu.cn
质　量　反　馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 刷 者：
装 订 者：
经　　销：全国新华书店
开　　本：185×260　　**印　张**：17.5　　**字　　数**：422 千字
版　　次：2010 年 11 月第 1 版　　**印　　次**：2010 年 11 月第1 次印刷
印　　数：1～　000
定　　价：　.00 元

产品编号：

出版说明

由于网络应用越来越普及，信息化的社会已经呈现出越来越广阔的前景，可以肯定地说，在未来的社会中电子支付、电子银行、电子政务以及多方面的网络信息服务将深入到人类生活的方方面面。同时，随之面临的信息安全问题也日益突出，非法访问、信息窃取、甚至信息犯罪等恶意行为导致信息的严重不安全。信息安全问题已由原来的军事国防领域扩展到了整个社会，因此社会各界对信息安全人才有强烈的需求。

信息安全本科专业是2000年以来结合我国特色开设的新的本科专业，是计算机、通信、数学等领域的交叉学科，主要研究确保信息安全的科学和技术。自专业创办以来，各个高校在课程设置和教材研究上一直处于探索阶段。但各高校由于本身专业设置上来自于不同的学科，如计算机、通信和数学等，在课程设置上也没有统一的指导规范，在课程内容、深浅程度和课程衔接上，存在模糊不清、内容重叠、知识覆盖不全面等现象。因此，根据信息安全类专业知识体系所覆盖的知识点，系统地研究目前信息安全专业教学所涉及的核心技术的原理、实践及其应用，合理规划信息安全专业的核心课程，在此基础上提出适合我国信息安全专业教学和人才培养的核心课程的内容框架和知识体系，并在此基础上设计新的教学模式和教学方法，对进一步提高国内信息安全专业的教学水平和质量具有重要的意义。

为了进一步提高国内信息安全专业课程的教学水平和质量，培养适应社会经济发展需要的、兼具研究能力和工程能力的高质量专业技术人才。在教育部相关教学指导委员会专家的指导和建议下，清华大学出版社与国内多所重点大学共同对我国信息安全人才培养的课程框架和知识体系，以及实践教学内容进行了深入的研究，并在该基础上形成了“信息安全人才需求与专业知识体系、课程体系的研究”等研究报告。

本系列教材是在课程体系的研究基础上总结、完善而成，力求充分体现科学性、先进性、工程性，突出专业核心课程的教材，兼顾具有专业教学特点的相关基础课程教材，探索具有发展潜力的选修课程教材，满足高校多层次教学的需要。

本系列教材在规划过程中体现了如下一些基本组织原则和特点。

(1) 反映信息安全学科的发展和专业教育的改革，适应社会对信息安全人才的培养需求，教材内容坚持基本理论的扎实和清晰，反映基本理论和原理的综合应用，在其基础上强调工程实践环节，并及时反映教学体系的调整和教学内容的更新。

(2) 反映教学需要，促进教学发展。教材要适应多样化的教学需要，正确把握教学内容和课程体系的改革方向，在选择教材内容和编写体系时注意体现素质教育、创新能

力与实践能力的培养，为学生知识、能力、素质协调发展创造条件。

（3）实施精品战略，突出重点。规划教材建设把重点放在专业核心（基础）课程的教材建设上；特别注意选择并安排一部分原来基础比较好的优秀教材或讲义修订再版，逐步形成精品教材；提倡并鼓励编写体现工程型和应用型的专业教学内容和课程体系改革成果的教材。

（4）支持一纲多本，合理配套。专业核心课和相关基础课的教材要配套，同一门课程可以有多本具有各自内容特点的教材。处理好教材统一性与多样化，基本教材与辅助教材、教学参考书，文字教材与软件教材的关系，实现教材系列资源的配套。

（5）依靠专家，择优落实。在制定教材规划时依靠各课程专家在调查研究本课程教材建设现状的基础上提出规划选题。在落实主编人选时，要引入竞争机制，通过申报、评审确定主编。书稿完成后认真实行审稿程序，确保出书质量。

繁荣教材出版事业，提高教材质量的关键是教师。建立一支高水平的、以老带新的教材编写队伍才能保证教材的编写质量，希望有志于教材建设的教师能够加入到我们的编写队伍中来。

21世纪高等学校信息安全专业规划教材

联系人：魏江江 weijj@tup.tsinghua.edu.cn

前　　言

随着计算机的广泛应用，以计算机为核心的信息系统安全问题越来越突出，如何采取有效的措施保护计算机系统安全，是目前各国都面临的问题。

本书具有完善的知识体系，概念清晰，讲解详细。本书具有以下特色：

(1) 内容全面，层次分明。从计算机系统的不同层次，介绍了各层次中的安全威胁及防范措施，覆盖了计算机系统安全涉及的各方面内容。主要包括系统安全(操作系统安全、数据库安全)、网络安全(漏洞检测、攻击与防范、防火墙、系统入侵检测与防御)、应用安全(电子邮件安全、IP 安全、Web 安全)。

(2) 强调信息保障与纵深防御。无论攻击和防守，对信息安全来说都是过程，信息安全是否成功关键在于对过程的把握。从纵深防御保护的角度分析了信息保障中的保护、检测、响应和恢复。如认证、防火墙起到边界保护的作用，扫描器、入侵检测系统、蜜罐则是检测工具，入侵防御系统属于响应工具。

(3) 注重理论与实践相结合，强调实用。在介绍理论知识的基础上，注重实践能力的培养，如第 5 章从理论到实践，介绍了漏洞的概念、漏洞的分类、漏洞标准库、漏洞扫描的原理、漏洞扫描的工具、扫描器的实现。

本书共分为 11 章：第 1 章是绪论；第 2 章介绍实现安全服务的各种方法；第 3 章与第 4 章分别介绍了操作系统安全、数据库安全；第 5 章介绍系统漏洞及检测；第 6 章介绍网络攻击与防范；第 7 章与第 8 章分别介绍了防火墙、系统入侵检测与防御；第 9～第 11 章介绍了电子邮件安全、IP 安全、Web 安全。其中第 3～第 5 章由李琳编写，第 9～第 11 章由黄石编写，其余各章由曹天杰编写，全书由曹天杰负责统稿。在本书的写作过程中，王宝忠、崔辉、杨珺涵、徐倩楠、冯倩、庞方参与了部分整理与校对工作，在此特别致谢。作者希望，通过学习这部教材，读者可以对计算机系统安全有一个全面深入的了解。

本书得到江苏省自然科学基金(No. BK2007035)、移动通信国家重点实验室开放研究基金(No. W200817)、中国矿业大学科技基金(No. 0D080309)的资助。

本书可以作为信息安全专业、信息对抗专业、计算机科学与技术专业、网络工程专业或其他相关专业的本科生和研究生教材，也可以作为信息系统安全领域的从业人员参考书。

由于作者水平有限，书中疏漏与错误之处在所难免，恳请广大同行和读者批评指正。作者的联系电子邮箱为：tjcao@cumt.edu.cn，请随时联系索取课程资料。

作　者

2010 年 3 月

目　　录

第1章　绪　　论

1.1　计算机安全的内涵

随着计算机的广泛应用，以计算机为核心的信息系统安全问题越来越突出。计算机系统安全涉及物理环境、计算机硬件、软件以及数据传输等各个方面。如何采取有效的措施保护计算机系统安全，是目前各国都面临的问题。

1.1.1　基本概念

计算机系统（computer system）也称计算机信息系统（computer information system），是由计算机及其相关的设备、设施（含网络）构成的，并按一定的应用目标和规则对信息进行采集、加工、存储、传输、检索等处理的人机系统。简言之，计算机系统是指信息系统赖以存在的实体和依赖于计算机实体所生成及运行的信息系统。

计算机系统安全中的“安全”一词指将计算机系统的服务和资源的任何弱点降到最低限度。计算机安全，是指为计算机系统建立和采取的技术与管理的安全保护措施，这些安全保护措施可以保护计算机系统中的硬件、软件以及数据，防止因偶然或恶意的原因而使系统或信息遭到破坏、更改或泄露。

计算机系统安全包括物理安全、运行安全以及信息安全三个方面。

1. 物理安全

保护计算机设备、设施（含网络）以及其他媒体免遭地震、水灾、火灾、有害气体和其他环境事故（如电磁污染等）破坏的措施和过程。特别是避免由于电磁泄漏产生信息泄露，从而干扰他人或受他人干扰。在信息系统安全中，物理安全是基础。如果物理安全得不到保证，如计算机设备遭到破坏，那么其他的一切安全措施就无从谈起。物理安全包括环境安全（机房与设施安全、环境与人员安全、自然灾害防护）、设备安全（防盗和防毁、防止电磁泄漏、防止线路截获、防电磁干扰、电源保护）和媒体安全（媒体数据安全、媒体本身安全）三方面。

2. 运行安全

为保障系统功能的安全实现，提供一套安全措施（如风险分析、审计跟踪、备份与恢复、应急等）来保护信息处理过程的安全。它侧重于保证系统正常运行，避免因为系统的崩溃和损坏而对系统存储、处理和传输的信息造成破坏和损失。运行安全包括风险分析、审计跟踪、备份与恢复、应急四个方面。

风险分析是指为了使计算机信息系统能安全地运行，首先了解影响计算机信息系统安全运行的诸多因素和存在的风险，从而进行风险分析，找出克服这些风险的方法。审计跟踪

是利用计算机信息系统所提供的审计跟踪工具,对计算机信息系统的工作过程进行详尽的跟踪记录,同时保存好审计记录和审计日志,并从中发现问题和及时解决问题,保证计算机信息系统安全可靠地运行。备份恢复与应急措施应同时考虑。根据所用信息系统的功能特性和灾难特点制定包括应急反应、备份操作、恢复措施三个方面内容的应急计划,一旦灾难事件发生,就可按计划方案最大限度地恢复计算机系统的正常运行。

3. 信息安全

信息是信息化社会中的重要资源。防止信息财产被故意地或偶然地非授权泄露、更改、破坏或防止信息被非法系统辨识和控制,避免攻击者利用系统的安全漏洞进行窃听、冒充、诈骗等有损合法用户利益的行为,本质上指保护用户的利益和隐私。

信息主要涉及信息传输的安全(动态安全)、信息存储的安全(静态安全)以及对网络传输信息内容的审计三方面。信息传输安全包括数据加密、数据完整性鉴别、防抵赖等;信息存储安全包括数据库安全、终端安全等;信息内容的审计是指实时对进出内部网络的信息进行内容审计,以防止或追查可能的泄密行为。

计算机信息系统安全的目的是保护计算机信息系统免遭偶发的或有意的非授权泄露、修改、破坏或丧失处理信息能力,实质是保护信息的安全,即机密性、完整性、可用性和不可否认性。

- 机密性:指信息不泄露给非授权实体并供其使用的特性。
- 完整性:指信息在存储和传输过程中未经授权不能被改变的特性。
- 不可否认性:指对出现的信息安全纠纷(如抵赖)提供调查的依据和手段的特性。
- 可用性:指信息能够被授权实体访问并按要求使用,信息系统能以人们所接受的质量水平持续运行,为人们提供有效的信息服务的特性。

以上属性均属于信息安全的基本属性,此外,还有一些属性也可用于描述信息安全的不同的特征,如可控性、可审计性、可生存性等。

基于信息安全的作用点,不同的应用会从不同的层面来关注信息安全的属性。面向数据的安全主要涉及保护信息的机密性、完整性、可用性和不可否认性;面向使用者的安全主要涉及认证、授权、访问控制、不可否认和可服务性以及基于内容的个人隐私、知识产权等的保护;面向管理者的安全,除上述层面外,还包括可控性。

1.1.2 计算机系统面临的攻击

一般把对信息系统的侵害行为称为攻击。计算机系统所面临的攻击大体上可以分为两类:一类是对实体的攻击,另一类是对信息的攻击。

对实体的攻击主要指对计算机及其外部设备和网络的攻击,如各种自然灾害与人为的破坏、设备故障、场地和环境因素的影响、电磁场的干扰或电磁泄露、战争的破坏、各种媒体的被盗和散失等。对信息系统实体的攻击,不仅会造成设备的损坏,而且会造成信息系统机密信息的泄露和破坏。

对信息的攻击大体可以分为两种:信息泄露和信息破坏。信息泄露是指偶然地或故意地获得(侦收、截获、窃取或分析破译)目标系统中的信息,特别是敏感信息。信息破坏是指

由于偶然事故或人为破坏，使信息的完整性和可用性受到破坏，如系统的信息被修改、删除、添加、伪造或非法复制，造成信息的破坏、修改或丢失。

影响计算机系统的因素很多，有些因素可能是有意的，也可能是无意的；可能是人为的，也可能是非人为的；可能是内部人为，也可能是外部攻击。

1．分类

1）被动攻击

根据攻击方法的不同，攻击可分为被动攻击和主动攻击两种。

被动攻击是指一切以窃密为目的的攻击，它的特点是偷听或监视传送，在不干扰系统正常工作的情况下侦收、截获、窃取系统信息，以便破译分析。例如利用观察信息、控制信息的内容来获得目标系统的位置和身份；利用信息长度和传递的频度获得信息的某些性质。被动攻击不容易被用户察觉出来，因此它的攻击持续性和危害性都很大。被动攻击的主要方法有：

（1）直接侦收。利用电磁传感器或隐藏的收发设备直接侦收或搭线侦收信息系统的中央处理机、外围设备、终端设备、通信设备或线路上的信息。

（2）截获信息。系统及设备在运行时，散射的寄生信号容易被截获。如离计算机显示终端百米左右仍可以接收到稳定、清晰可辨的信息图像。此外，短波、超短波、微波和卫星等无线电通信设备有相当大的辐射量，市话线路、长途架空明线等电磁辐射也相当严重，因此可截获系统设备的电磁辐射信息。

（3）合法窃取。利用合法用户身份，设法窃取未被授权的信息。例如，在统计数据库中，利用多次查询数据的合法操作，推导出不该了解的机密信息。

（4）破译分析。对于已经加密的信息，利用各种破译分析手段，获得机密信息。

（5）从遗弃的媒体中分析获取信息。如从遗弃的打印纸、各种记录和统计报表、窃取或丢失的软盘片中获得有用信息。

2）主动攻击

主动攻击是指篡改信息的攻击，它不仅能窃密，而且还威胁到信息的完整性。主动攻击有选择地修改、删除、添加、伪造和重排信息内容，造成信息破坏。主动攻击的主要方法有以下几种。

（1）窃取并干扰通信线路中的信息。

（2）返回渗透。有选择地截获系统中央处理机的通信，然后将伪信息返回给系统用户。

（3）线间插入。当合法用户已占用信道，但是终端设备还没有动作时，插入信息进行窃听或信息破坏活动。

（4）非法冒充。采取非常规的方法和手段，窃取合法用户的标识符，冒充合法用户进行窃取或信息破坏。

（5）系统人员的窃密和毁坏系统数据、信息的活动等。

主动攻击具有与被动攻击相反的特点。虽然很难检测出被动攻击，但可以采取措施防止被动攻击。相反，很难预防主动攻击，因为这样需要随时对所有的通信工具和路径进行完全的保护。防止主动攻击的做法是对攻击进行检测，并从它引起的中断或延迟中恢复过来，因为检测具有威慑的效果。

2. 方法

下面列举几种对信息系统常见的攻击方法。

1）暴力攻击

运用计算和网络资源尝试密码的每一种可能组合，称为暴力攻击。因为这常常是重复猜测用户账户的密码，有时也称为密码攻击。

2）拒绝服务攻击

在拒绝服务攻击中，攻击者给目标发送很多连接或信息请求，目标系统无法成功地处理如此多的请求以及其他合法的服务请求，这将引起系统崩溃或者无法执行日常功能。任何连接到因特网并提供基于 TCP 网络服务的系统（比如 Web 服务器、FTP 服务器或邮件服务器）都是拒绝服务攻击的潜在目标。

3）嗅探

嗅探器是指能够监视在网络上传输的数据的程序或者设备。嗅探器可用于合法的网络管理，也可用于从网络上窃取信息。未授权的嗅探器对于网络安全来说是非常危险的，因为他们实际上不可能被发现，但几乎可嵌入任何地方。嗅探器经常工作在 TCP/IP 的网络上，它们有时称为数据包嗅探器。嗅探器给网络增加了危险，因为许多系统和用户在本地网络上通过明文发送信息。嗅探器可以显示出所有通过的数据，包括口令和文件中的数据。

4）欺骗

欺骗是一种用来非法访问计算机的技术。例如攻击者要进行 IP 地址欺骗，会首先找到一台可信赖的主机的 IP 地址，然后更改数据包（如图 1-1 所示），使这个数据包看上去来自这台可信赖的主机。

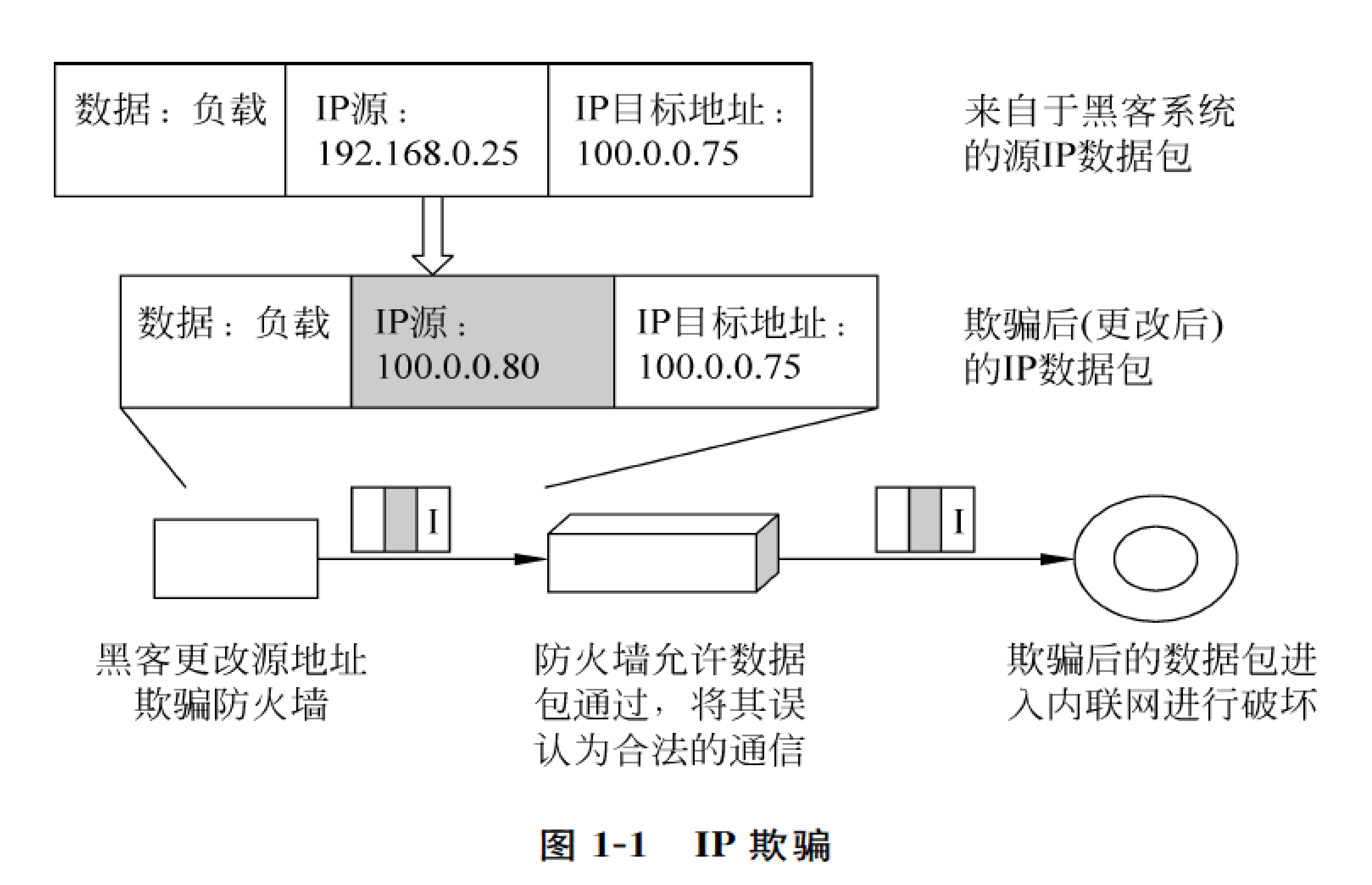


图 1-1 IP 欺骗

5）社会工程

在信息安全中，社会工程是指使用社交技能欺骗人们。比如，攻击者打电话给公司的总机，问出总经理的名字，然后给公司其他人打电话时提及总经理的名字，抬高自己的身份（虽然是假的），以便获得更多的信息。有时攻击者使用威胁、哄骗或者乞求等手段来动摇目标对象。

1.2　安全策略、机制与服务

1.2.1　安全策略

安全策略是指在一个特定的环境里(安全区域),为了保证提供一定级别的安全保护所必须遵守的一系列条例和规则。例如,可以将安全策略定义为:系统中的用户和信息被划分为不同的层次,一些级别比另一些级别高。当且仅当主体的级别高于或等于客体的级别,主体才能读访问客体;当且仅当主体的级别低于或等于客体的级别,主体才能写访问客体。

一种安全策略实质上表明所涉及的系统在进行一般操作时,在安全范围内什么是允许的,什么是不允许的。策略通常不作具体规定,它只是提出什么是最重要的,而不确切地说明如何达到所希望的这些结果。

安全策略都建立在授权的基础之上,一般按授权性质的不同区分不同的策略。在安全策略中包含对"什么构成授权"的说明。在一般性的安全策略中可能写有"未经适当授权的实体,信息不得给予、不被授权、不允许引用、任何资源也不得为其使用"。

按照所涉及的授权的性质可将策略分为三种,基于规则的策略、基于身份的策略、基于角色的策略。

基于身份的安全策略使用建立在不多的一般属性或敏感类之上的规则,它们通常是强加的。它的基础是用户的身份和属性以及被访问的资源或客体的身份和属性。在一定程度上与"必须认识"的安全观念相当。它的目的是过滤对数据或资源的访问。基本上有两种执行基于身份的策略的方法,将有关访问权的信息视为访问者所拥有,或是视为被访问数据的一部分。前者的例子为特权标识或权力,给予用户并为代表该用户进行活动的进程所使用,后者的例子为访问控制表。这两种情况下,数据项的大小可以有很大的变化(从完整的文件到数据元素),这些数据项可以按权力命名,或带有它自己的访问控制表。

基于规则的安全策略涉及建立在特定的、个体化属性之上的授权准则,假定某些属性与被应用实体永久相关联,而其余属性可以是某种占有物(如权力),它们可传送给另外的实体。它的基础是强加于全体用户的总安全策略,是为了以最小的代价,保证信息系统的安全,使信息系统发挥最大的效益。基于规则的安全策略中的授权通常依赖于敏感性。在一个安全系统中,数据或资源应该标注安全标记。代表用户进行活动的进程可以得到与其原发者相应的安全标记。

基于角色的安全策略按角色使用授权机制,它为每个个体分配角色,按角色分配许可。

1.2.2　安全机制

安全策略确定以后,需要由不同的安全机制来实施。安全机制是实施安全策略的方法、工具或者章程。如果给定对"安全"和"非安全"行为进行描述的安全策略规范,安全机制就能够阻止攻击,检测攻击,或者遭到攻击后恢复工作。例如:假如安全策略设定为禁止任何学生复制其他学生的作业文件。计算机系统提供防止其他人阅读用户文件的机制。Anna没有使用这些机制来保护她的作业文件,并且被 Bill 复制了作业。这就发生了安全破坏,因

为 Bill 违反了安全策略。Anna 未能保护好她的文件，但她并未授权 Bill 复制她的作业。安全机制可单独实施，也可以组合使用。通常有三类安全机制：预防、检测和恢复。典型的安全机制有以下几种。

1. 加密机制

通过加密、解密，不仅可以实现数据安全存储和安全传输，也可以实现身份鉴别、数据完整性、不可否认性等特性，从而保证信息的安全。采用加密技术，可以有效地防止信息的未授权读取和修改，抵赖、否认和伪造，以及通信业务流分析。

2. 数字签名机制

数字签名机制用于实现抗抵赖和不可否认服务及鉴别对方身份真实性等特殊服务的场合。数字签名机制的主要特征是：不使用秘密密钥就不能建立签名数据单元。这说明数字签名具有以下特征：除了掌握秘密密钥的人以外，任何人都不能建立数字签名；接收者不能建立签名数据单元；发送者不能否认曾经发送过签名的数据单元。

3. 访问控制机制

访问控制机制实施对资源的访问加以限制的策略，即规定不同主体对不同客体的操作权限，目的是只允许被授权用户访问敏感资源，拒绝未授权用户的访问。可采用的技术有访问控制矩阵、口令、能力表等，它们可以说明用户的访问权限。

4. 数据完整性机制

数据完整性机制保护单个数据单元和整个数据单元的完整性，包括各种信息流错误检验、校验等技术。

5. 鉴别交换机制

数据鉴别交换机制指信息交换双方之间的相互鉴别。当交换信息的双方和通信手段均可信任时，可通过口令来鉴别；当交换信息双方可信，而通信手段不可靠时，可由密码技术加以保护；当交换信息双方互不信任时，可使用数字签名等技术来实现抗抵抗服务。

6. 路由选择机制

通过路由选择机制，可保证数据只在物理上安全的路由上传输，保证机密信息只在具有适当保护措施的路由上传输。

7. 抗信息流分析机制

该机制通过产生伪通信业务，将数据单元通信量填充到预定的数量，以防止通过通信业务流分析获取业务流量信息。在具体实施时，应注意加密和伪装，以避免非法用户区分出伪通信业务和真正的通信业务。

8. 公证机制

在通信过程中，信息的完整性、信源、通信时间和目的地、密钥分配、数字签名等，均可以借助公证机制加以保证。保证是由第三方公证机制提供，它接受通信实体的委托，并掌握可供证明的可信赖的所需信息。公证可以是仲裁方式或判决方式的。

9. 环境安全机制

物理机制的安全是保证信息安全的重要措施。硬件、软件和通信安全的第一道安全屏

障就是物理环境的保护，必须有适应不同策略要求的保护措施。

10. 审查与控制机制

各种各样的防护措施均离不开人的掌握和控制，因此，系统的安全最终是由人来控制的。安全离不开人员的审查、控制和管理，要通过制定、执行和实施各种管理制度以及各种安全保护条例来实现。

1.2.3 安全服务

安全服务(security service)是由安全机构所提供的主要的安全功能。安全机构是实现和执行各种安全策略的功能的集合。安全策略是提供安全服务的一套准则。一项安全服务可以由若干项安全机制来实现。安全服务提供者对信息的机密性、完整性和来源的真实性进行保护和鉴别的过程称为安全服务。

针对网络系统受到的威胁，OSI 安全体系结构提出了 5 类安全服务。

1. 认证服务

认证服务提供通信对等实体认证和数据来源认证。对等实体认证用于开放系统的同等层中的实体在建立连接或数据传输阶段，对对方实体(用户或进程)的合法性、真实性进行确认，以防止假冒。认证可以是单向或双向的。数据来源认证服务用于对数据单元的来源提供确认，证明某数据与某实体有着静态不可分的关系。但它对数据单元的重复不提供认证保护。

2. 访问控制服务

访问控制服务可以防止未经授权的用户非法使用系统资源。这种服务不仅可以提供给单个用户，也可以提供给封闭用户组中的所有用户。在用户身份得到认证和授权以后，访问控制服务将根据预先设定的规则对用户访问某项资源进行控制，只有规则允许时才能访问，违反预定的安全规则的访问行为将被拒绝。

3. 数据机密性服务

数据机密性服务的目的是保护网络中各系统之间交换的数据，防止因数据被截获而造成的泄密。加密可向数据或业务流信息提供保密性，并且可以对其他安全机制起作用或对它们进行补充。

4. 数据完整性服务

这种服务用来防止非法实体对用户的主动攻击，以保证数据接收方收到的信息与发送方发送的信息完全一致。

确定单个数据单元的完整性涉及两个处理，一个在发送实体时进行，一个在接收实体时进行。发送实体给数据单元附加一个分组校验码或密码校验值之类的补充信息，而且它本身可以被加密。接收实体进行验证比较，确定该数据在传输过程中是否被篡改。

5. 不可否认性服务

这种服务有两种形式：第一种是源发证明，为数据的接收者提供数据来源的证据，它用来防止发送数据方发送数据后否认自己发送过数据。第二种形式是交付证明，为数据的发送者提供数据交付证据，它用来防止接收方接收数据后否认自己收到过数据。

1.3 纵深防御

不论攻击和防守，对信息安全来说都是过程，信息安全是否成功关键在于对过程的把握。在这个过程中设置的防御层数越多，对系统资源进行未授权访问的难度就越大。我们通过提供冗余防御层来确保安全性，在某一层或者在某些情况下多个层被攻破时，冗余防御层能够对资源进行保护。

1.3.1 纵深防御体系的概念

安全体系不应只依靠单一安全机制和多种安全服务的堆砌，应该建立互相支撑的多种安全机制，建立具有协议层次和纵向结构层次（信息流方向）的完备体系，通过多层机制互相支撑来获取整个信息系统的安全。这种原则在日常生活中也是常见的，例如大门上既安装了门锁，又安装了插销；汽车要开车门还要开点火器才能发动等。

纵深防御（defense-in-depth）最初来源于军事战略，有时也称为弹性防御或是深层防御，是以全面深入的防御去拖延而不是阻止前进中的敌人，通过放弃空间来换取时间而抵御敌人。有别于通过一个单一而强大的防御战线去防御敌人，纵深防御试图用下列方式阻击敌人：（1）使攻击潮流在一段时间内失去动能；（2）使攻击方在一个广大地区内失去攻击能力。

对防御者而言，他会放弃在领土上相对微弱的抵抗，全力压迫攻击方的后勤补给，或是切割敌方在数量上的优势的兵力。一旦攻击方失去其动能，或是在大部分地区被切割后的兵力数量优势不再，防御反攻将发动。其主旨在于促使敌方资源的消耗，进而带动消耗战，或是假使攻击方退回原本攻击起始点。

纵深防御思想是近年来才发展起来的，它从各个层面（包括主机、网络、系统边界和支撑性基础设施等），根据信息资产保护的不同等级来保障信息与信息系统的安全，实现预警、保护、检测、反应和恢复这五个安全内容。纵深防御是美国国防部提出的信息保障策略，该策略的基本原理可应用于任何机构的信息系统或网络之中。

纵深防御体系就是将分散系统整合成一个异构网络系统，基于联动联防和网络集中管理、监控技术，将所有信息安全和数据安全产品有机地结合在一起，在漏洞预防、攻击处理、破坏修复三方面给用户提供整体的解决方案，能够极大地提高系统防护效果，降低网络管理的风险和复杂性。同时由于黑客攻击的方式具有高技巧性、分散性、随机性和局部持续性的特点，因此即使是多层面的安全防御体系，如果是静态的，也无法抵御来自外部和内部的攻击，只有将众多的攻击手法进行搜集、归类、分析、消化、综合，将其体系化，才有可能使防御系统与之相匹配、相符合，以自动适应攻击的变化，从而形成动态的安全防御体系。

1.3.2 网络防御模型

信息保障技术框架（Information Assurance Technical Framework，IATF）由美国国家安全局 NSA 起草颁布。在 IATF 中定义了对一个系统进行信息保障的过程以及系统中所有部件的安全需求。遵循这些原则，信息保障被分成 4 个部分：保护（protect）、检测

(detect)、响应(respond)和恢复(restore),即所谓的 PDRR 模型,如图 1-2 所示。

- 保护:通过传统的静态安全技术和方法可用来实现保护的环节,包括系统加固、防火墙、加密机制、访问控制和认证等。
- 检测:检测在 PDRR 模型中占据着重要的地位,它是动态响应和进一步加强保护的依据,也是强制落实安全策略的有力工具。只有检测和监控信息系统(通过漏洞扫描和入侵检测等手段),及时发现新的威胁和漏洞,才能在循环反馈中作出有效的响应。
- 响应:响应和检测环节是紧密关联的,只有对检测中发现的问题作出及时有效的处理,才能将信息系统迅速调整到新的安全状态,或者叫最低风险状态。
- 恢复:恢复环节对于信息系统和业务活动的生存起着至关重要的作用,组织只有建立并采用完善的恢复计划和机制,其信息系统才能在重大灾难事件中尽快恢复并延续业务。

在这个模型的基础上,有人提出在前面加上预警(waring),在后面加上反击(counterattack),形成如图 1-3 所示的层次结构。预警是指根据对已经发生的网络攻击或正在发生的网络攻击及其趋势的分析,以及对本地网络的安全性分析,对可能发生的网络攻击提出警告。反击是指遭受到网络攻击后,除了及时阻止网络攻击外,还要及时恢复遭到破坏的本地系统,并及时地对外提供正常的服务。

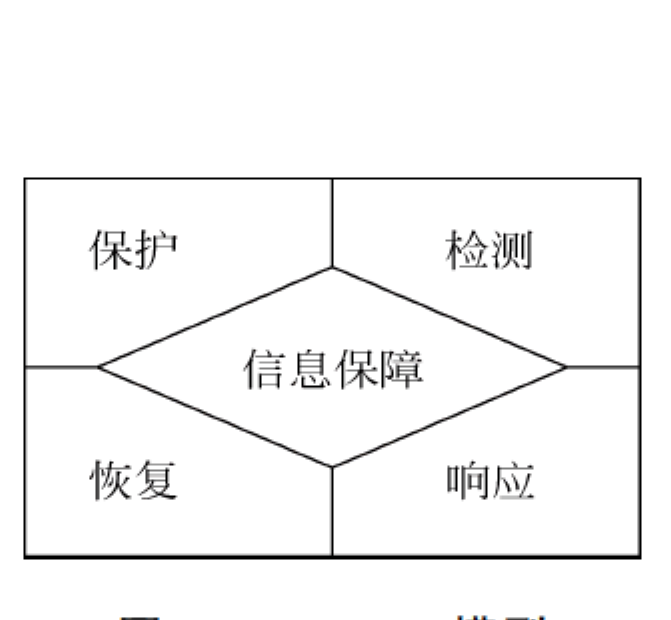


图 1-2　PDRR 模型

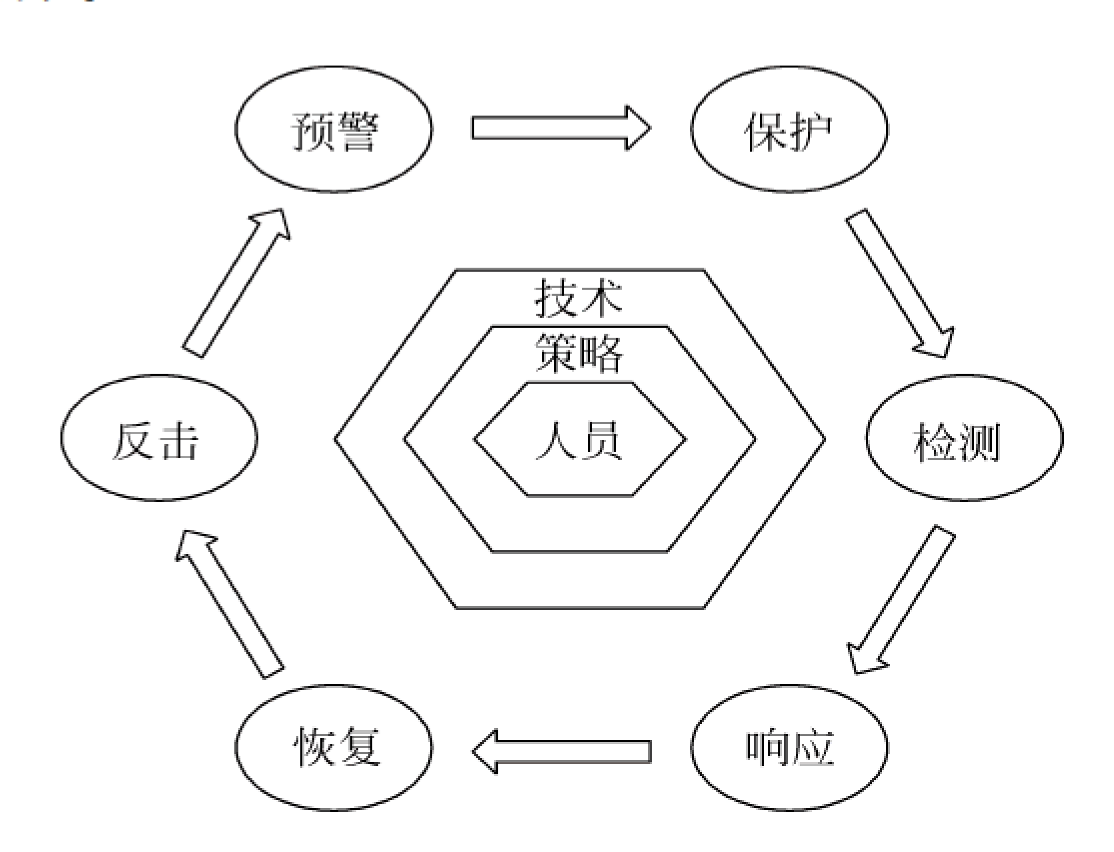


图 1-3　WPDRRC 模型

该模型全面涵盖了各个安全因素,突出了人、策略、管理的重要性,反映了各个安全组件之间的内在联系。这三个要素相互依存,相互配合,相互支撑,相互制约,相互之间具有层次和互动关系。

人:包括用户、运行管理员(系统管理员、网络管理员等)、安全管理员(安全管理员、密钥管理员等)、安全审计管理员等;这是整个体系中最活跃、最能动、最主要的因素,是主导者、操作者。

管理:包括技术管理(应用、研发、运行、安全保密测评、应急响应等)、行政管理(组织、认证、认可、应急处置、法律法规、标准、规范规程、规章制度、安全策略制订、宣传教育、培训等)。

许多信息安全问题是由于管理不善造成的,而这些安全问题大多可以通过科学的信息

安全管理来避免。因此,管理已成为信息安全保障能力的重要基础。目前,信息安全管理已成为世界信息安全研究领域的重点和难点,信息安全管理标准也成为信息安全标准体系的重要组成部分。

技术:各种安全防护技术(物理隔离、防火墙、VPN、身份认证、访问控制、入侵检测、漏洞扫描、加密、安全审计、病毒防护、非法外连监控、操作系统安全、数据库安全、数据备份、灾难恢复及应急响应等)。信息安全保密技术是不断发展、不断完善、不断涌现的。并且呈现出从离散、单一、独立的各个技术产品向产品功能集成、各类产品联动发展的趋势。

1.4 信息系统安全保护等级划分准则

信息系统安全等级保护应依据信息系统的安全保护等级情况,保证它们具有相应等级的基本安全保护能力,不同安全保护等级的信息系统要求具有不同的安全保护能力。

国家标准 GB 17859—1999《计算机信息系统安全保护等级划分准则》是开展等级保护工作的基础性标准,是信息安全等级保护系列标准编制、系统建设与管理、产品研发、监督检查的科学技术基础和依据。这个标准将计算机信息系统的安全性从低到高划分为 5 个等级,分别为用户自主保护级、系统审计保护级、安全标记保护级、结构化保护级和访问验证保护级,高级别安全要求是低级别要求的超集。随着安全保护等级的提高,计算机信息系统安全保护能力逐渐增强。

1.4.1 第一级 用户自主保护级

用户自主保护级的计算机信息系统可信计算基(Trusted Computing Base,TCB)通过隔离用户与数据,使用户具备自主安全保护的能力。它具有多种形式的控制能力,对用户实施访问控制,即为用户提供可行的手段,保护用户和用户组信息,避免其他用户对数据的非法读写与破坏。可信计算基是指计算机系统内保护装置的总体,包括硬件、固件、软件和负责执行安全策略的组合体,它建立了一个基本的保护环境并提供一个可信计算系统所要求的附加用户服务。

本级实施的是自主访问控制,即通过 TCB 定义系统中的用户和命名用户对命名客体的访问。在本安全级中,应允许命名用户以用户或用户组的身份规定客体的共享权限并实施访问控制,并阻止非授权用户读取敏感信息。这意味着系统用户或用户组可以通过 TCB 自主地定义主体对客体的访问权限。

本级别为用户提供身份鉴别,即在系统初始化时,TCB 首先要求用户标识自己的身份,然后使用保护机制(例如口令)来鉴别用户的身份,并实施对客体的自主访问控制,避免"非法"用户数据的读写或破坏。

从数据完整性角度看,TCB 通过自主完整性策略,阻止非授权用户修改或破坏敏感信息。

1.4.2 第二级 系统审计保护级

与用户自主保护级相比,系统审计保护级的计算机信息系统 TCB 实施了粒度更细的自

主访问控制,通过登录规程、审计与安全相关的事件和隔离资源等措施,使用户对自己的行为负责。

本级实施的是自主访问控制和客体(object,信息的载体)的安全重用。在自主访问控制方面,TCB 实施的自主访问控制粒度是单个用户,并控制访问控制的扩散,即没有访问权的用户只允许由授权用户指定其对客体的访问权。在客体的安全重用方面,客体被初始指定或分配给一个主体之前,或在客体再分配之前,必须撤销该客体所含信息的授权;当一个主体(subject,引起信息在客体之间流动的人、进程或设备等)获得一个客体的访问权时,对当前主体而言,原主体的活动所产生的任何信息都是不可获得的。

从用户的角度来看,系统审计保护级有两个功能:身份鉴别和安全审计。

本级的身份鉴别比用户自主保护级增加两点:(1)为用户提供唯一的标识,确保用户对自己的行为负责;(2)为支持安全审计功能,具有将身份标识与用户所有可审计的行为相关联的能力。

计算机信息系统 TCB 能创建和维护受保护客体的访问审计跟踪记录,并能阻止非授权的用户对它访问或破坏。TCB 能记录下述事件:使用身份鉴别机制;将客体引入用户地址空间(例如:打开文件、程序初始化);删除客体;由操作员、系统管理员或(和)系统安全管理员实施的动作,以及其他与系统安全有关的事件。对于每一事件,其审计记录包括:事件的日期和时间、用户、事件类型、事件是否成功。对于身份鉴别事件,审计记录包含的来源(例如:终端标识符);对于客体引入用户地址空间的事件及客体删除事件,审计记录包含客体名。对不能由 TCB 独立分辨的审计事件,审计机制提供审计记录接口,可由授权主体调用。这些审计记录区别于计算机信息系统可信计算基独立分辨的审计记录。

在数据完整性方面,TCB 应提供并发控制机制,以确保多个主体对同一客体的正确访问。

1.4.3　第三级　安全标记保护级

安全标记保护级的计算机信息系统 TCB 除具有系统审计保护级的所有功能外,还提供有关安全策略模型、数据标记以及主体对客体强制访问控制的非形式化描述,具有准确地标记输出信息的能力,可消除通过测试发现的任何错误。

本级的计算机信息系统 TCB 定义和控制系统中命名用户对命名客体的访问。实施机制(如访问控制表)允许命名用户以用户或用户组的身份规定并控制客体的共享;阻止非授权用户读取敏感信息。并控制访问权限扩散。自主访问控制机制根据用户指定方式或默认方式,阻止非授权用户访问客体。访问控制的粒度是单个用户。没有访问权的用户只允许由授权用户指定对客体的访问权。阻止非授权用户读取敏感信息。

TCB 对所有主体及其所控制的客体(例如:进程、文件、段、设备)实施强制访问控制,为这些主体及客体指定敏感标记(sensitivity label,表示客体安全级别并描述客体数据敏感性的一组信息,可信计算基中把敏感标记作为强制访问控制决策的依据),这些标记是等级分类和非等级类别的组合,它们是实施强制访问控制的依据。计算机信息系统可信计算基支持两种或两种以上成分组成的安全级。计算机信息系统可信计算基控制的所有主体对客体的访问应满足:仅当主体安全级中的等级分类高于或等于客体安全级中的等级分类,且主体安全级中的非等级类别包含了客体安全级中的全部非等级类别,主体才能读客体;仅

当主体安全级中的等级分类低于或等于客体安全级中的等级分类，且主体安全级中的非等级类别包含了客体安全级中的非等级类别，主体才能写一个客体。

敏感标记是实施强制访问的基础，TCB 应维护与主体及其控制的存储客体（例如：进程、文件、段、设备）相关的敏感标记。为了输入未加安全标记的数据，计算机信息系统 TCB 向授权用户要求并接受这些数据的安全级别，且可由计算机信息系统 TCB 审计。

从用户角度看，此级也包括身份鉴别和安全审计两大功能。TCB 除了具有第二级的功能外，还有以下功能：

（1）确定用户的访问权限和授权数据；

（2）接受数据的安全级别，维护与每个主体及其控制下的存储对象相关的敏感标记；

（3）维护标记的完整性；

（4）维护并审计标记信息的输出，并与相关联的信息进行匹配；

（5）确保以该用户的名义而创建的那些在 TCB 外部的主体和授权，受其访问权和授权的控制。

对于数据完整性，TCB 通过自主和强制完整性策略，阻止非授权用户修改或破坏敏感信息。在网络环境中，使用完整性敏感标记来确信信息在传送中未受损。

1.4.4 第四级 结构化保护级

结构化保护级的计算机信息系统 TCB 建立于一个明确定义的形式化安全策略模型之上，它要求将第三级系统中的自主和强制访问控制扩展到所有主体与客体。此外，还要考虑隐蔽通道（covert channel，允许进程以危害系统安全策略的方式传输信息的通信信道）。本级的计算机信息系统 TCB 必须结构化为关键保护元素和非关键保护元素。计算机信息系统 TCB 的接口也必须明确定义，使其设计与实现能经受更充分的测试和更完整的复审。此外，本级还加强了鉴别机制；支持系统管理员和操作员的职能；提供可信设施管理；增强了配置管理控制。因此，系统具有相当的抗渗透能力。

本级的主要特征如下：

（1）TCB 基于一个明确定义的形式化安全保护策略。

（2）将第三级实施的（自主和强制）访问控制扩展到所有主体和客体。在自主访问控制方面，TCB 应维护由 TCB 外部主体直接或间接访问的所有资源的敏感标记；在强制访问控制方面，TCB 应对所有可被其外部主体直接或间接访问的资源实施强制访问控制，应为这些主体和客体指定敏感标记。

（3）针对隐蔽信道，将 TCB 构造成为关键保护元素和非关键保护元素。

（4）TCB 具有合理定义的接口，使其能够经受严格的测试和复查。

（5）通过提供可信路径来增强鉴别机制。

（6）支持系统管理员和操作员的可确认性，提供可信的实施管理方法，增强严格的配置管理控制。

在审计方面，当发生安全事件时，TCB 还能够检测事件的发生，记录审计条目，通知系统管理员，标识并审计可能利用隐蔽信道的事件。

在隐蔽信道分析方面，系统开发者应彻底搜索隐蔽存储信道，并根据实际测量或工程估算确定每一个被标识信道的最大带宽。

1.4.5　第五级　访问验证保护级

访问验证保护级的计算机信息系统可信计算基满足访问监控器需求。访问监控器本身是抗篡改的；必须足够小，能够分析和测试。为了满足访问监控器需求，计算机信息系统可信计算基在其构造时，排除那些对实施安全策略来说并非必要的代码；在设计和实现时，从系统工程角度将其复杂性降低到最小程度。支持安全管理员职能；扩充审计机制，当发生与安全相关的事件时发出信号；提供系统恢复机制。系统具有很高的抗渗透能力。

本级与第四级相比，主要区别有以下四个方面。

(1) 在 TCB 的构造方面，具有访问监控器。所谓访问监控器(reference monitor)，就是监控主体与客体之间授权访问关系的部件，仲裁主体对客体的全部访问。访问监控器必须是抗篡改的，并且是可分析、可测试的。

(2) 在自主访问控制方面，因为有访问控制器，所以访问控制能够为每个客体指定用户和用户组，并规定他们对客体的访问模式。

(3) 在审计方面，在访问监控器的支持下，TCB 扩展了审计能力。本级的审计机制能够监控可审计安全事件的发生和积累，当积累超过规定的门限值时，能立即向系统管理员发出报警；并且，如果这些与安全相关的事件继续发生，能以最小的代价终止它们。

在系统的可信恢复方面，TCB 提供了一组过程和相应的机制，保证系统失效或中断后，可以在不损害任何安全保护性能的情况下进行恢复。

习　题　1

1. 什么是计算机系统安全?
2. 对于计算机系统安全，有哪些典型的攻击? 简单列举几个。
3. 实施纵深防御的技术有哪些?
4. 什么是系统安全策略?
5. OSI 安全体系结构中，包括哪些安全服务? 安全服务与安全机制之间的关系如何?
6. 结构化保护级的主要特征有哪些?

第 2 章　实现安全服务

2.1　认　　证

认证服务是用来识别用户身份的服务。用户身份,特别是远程用户很难识别他们的身份,尤其是那些想要进行破坏的用户,他们可能冒充成合法的用户。认证服务提供给系统一种验证用户身份的能力。认证是一种最重要的安全服务,因为在某种程度上所有其他安全服务都依赖于它。

2.1.1　认证概述

认证是用于一种特定的通信过程,在此过程中需要提交实体的身份。认证又分为以下两种情况。

(1) 实体认证。身份是由参与某次通信或会话的远端的一方提交的。这种认证只是简单地认证实体本身的身份,不会和实体想要进行的任何活动联系起来。显然,它的作用是有限的。因此,在实际中,实体认证通常会产生一个明确的结果,允许实体进行其他活动或通信。

(2) 数据源认证。身份是由声称它是某个数据项的发送者的那个实体所提交的。此身份连同数据项一起发送给接收者。这种认证就是认证某个特定的数据项是否来源于某个特定的实体,为了确定被认证实体与一些特定数据项有着静态的不可分割的联系。

认证系统的设计目的是允许授权用户进入,阻止未授权用户进入,这涉及三个概念:鉴别、认证和授权。

(1) 鉴别:鉴别步骤要求用户通过一个令牌或者标识串(如 E-email 地址、账户号码)来确认自己的身份。

(2) 认证:一旦鉴别标志被出示,在认证步骤中用户必须提供证明他身份的证据。

(3) 授权:授权允许一个认证用户在获得许可后进行一系列操作。

人们通过他们知道的或能够识别的事物、持有的物件、自身的生物特征等来进行自我认证。前两种情况下,用户和系统共享一个秘密(如认证密钥)。在注册阶段,用户和系统商议这个秘密;在认证阶段,系统确定认证的用户是否拥有事先商议的秘密。如果用户证明了秘密的知识,那么系统确定了用户的身份。第三种情况下,系统在注册阶段记录用户身体或行为特征的一个数字描述,并在认证阶段进行确认。

许多认证通过使用公钥密码学来加强,比如,用户可能有一张包含公钥和相应的私钥的智能卡。用户的公钥取代口令存储在远程计算机系统的文件中。为了对用户进行认证,远程计算机发送给用户一个随机数;用户用他的私钥对该随机数进行签名,并发送签名结果给远程服务器,远程服务器用文件中的公钥验证签名。通过这种方法,远程系统不必收到公钥就能够验证用户拥有私钥。

一种基于公钥密码学的认证系统中,用户必须通过口令、生物特征、私有物品等向本地系统(智能卡)认证他自己,然后智能卡使用公钥密码体制向远程系统认证它自己。因为远程系统相信智能卡是可靠的,远程系统相信智能卡的主体已经被认证。这就是一个传递信任(transitive trust)的例子。

图 2-1 描述了认证过程中涉及的实体。在该过程的每个阶段,攻击者都可能访问认证密钥。密码技术能够有效地保护网络中密钥的传递,服务器端处理密钥,密钥存储在文件存储器中。真正的漏洞是输入机制和用户。基于知识的认证中,用户必须持有一个密钥,密钥可以被告诉、发现、窃取。用户可能告诉其他人他们的密钥,因为他们可能被欺骗或被强制这样做,或者因为他们不理解共享密钥可能产生的后果。即使用户不在认证前揭示密钥,密钥仍可能被发现,比如用户输入密钥的过程被监视了,或者被攻击者根据输入机制推导出密钥。

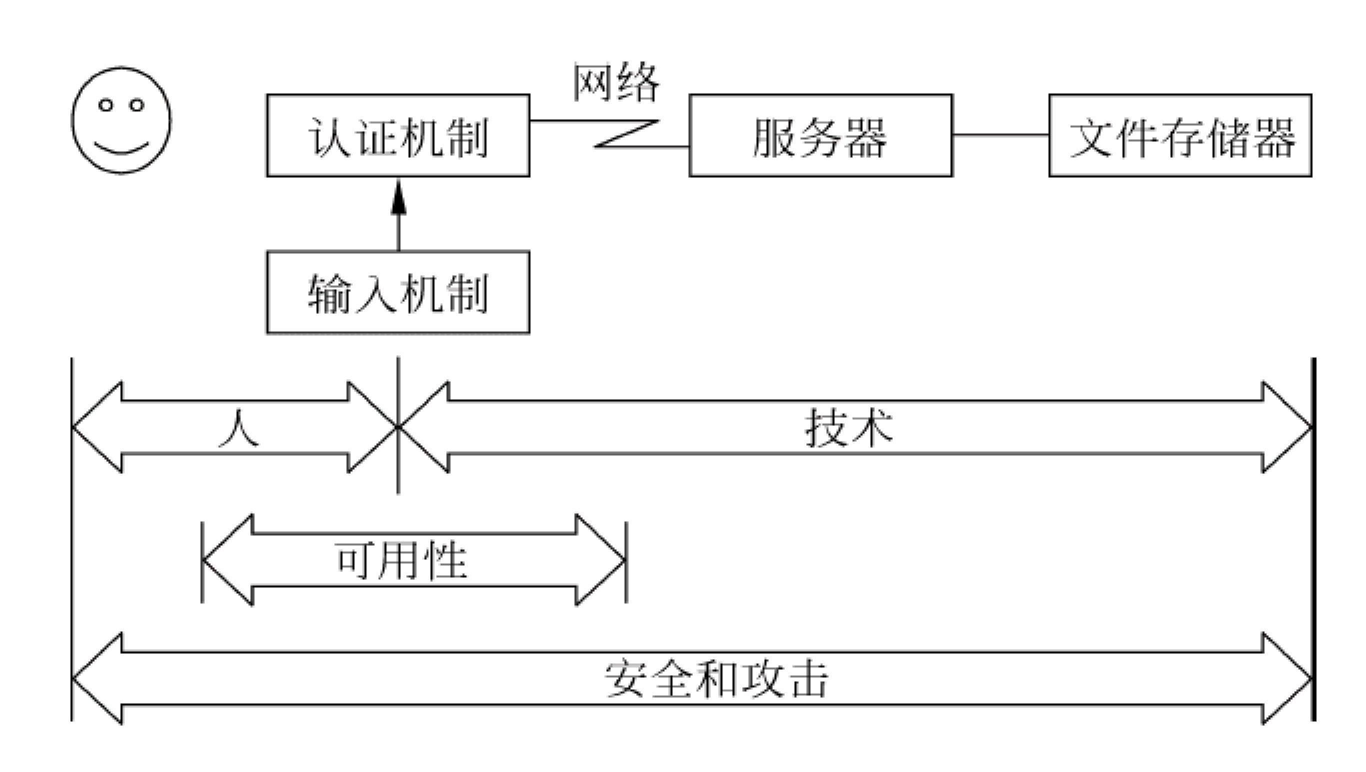


图 2-1　认证过程中的实体

安全不能通过纯技术的方法解决,例如用户访问障碍。

身体的:一些认证机制依赖于身体的能力来使用鼠标,这严重影响了有各种身体疾病的用户。

认知的:这种无能力是指阅读障碍、运动障碍、记忆力衰退等。阅读障碍影响用户工作与文本的能力,运动障碍影响用户按顺序记忆和执行一系列指令的能力,记忆力衰退的用户会发现很难正确的记起基于知识的认证密钥。

知觉的:举一个例子,依赖于视力的认证机制是一个难题,不仅是盲人,老年人也有这样的问题。

技术的:举一个例子,大量数据进行交换的认证机制可能给通过调制解调器访问系统的用户带来不便。与此相似,那些要求用户识别图像的认证机制可能使得通过移动设备访问系统的用户无法访问。

2.1.2　生物特征识别

所谓生物特征识别,是指通过计算机与各种传感器和生物统计学原理等高科技手段的密切结合,利用人体固有的生理特性和行为特征,来进行个人身份鉴定的方法。生理特征与生俱来,多为先天性的;行为特征则是习惯使然,多为后天性的。将生理和行为特征统称为生物特征。

并非所有的生物特征都可用于个人的身份鉴别。身份鉴别可利用的生物特征必须满足以下几个条件。

普遍性：即必须每个人都具备这种特征。

唯一性：即任何两个人的特征是不一样的。

可测量性：即特征可测量。

稳定性：即特征在一段时间内不改变。

当然，在应用过程中，还要考虑其他实际因素，比如识别精度、识别速度、对人体无伤害、被识别者的接受性等。生物特征识别可以分为两类。

1. 生理特征识别

生理特征识别包括人脸识别、虹膜识别、手形识别、指纹识别、掌纹识别等。

人脸识别：人脸识别技术通过对面部特征和它们之间的关系（眼睛、鼻子和嘴的位置以及它们之间的相对位置）来进行识别，用于捕捉面部图像的两项技术为标准视频和热成像技术。标准视频技术通过视频摄像头摄取面部的图像，热成像技术通过分析由面部的毛细血管的血液产生的热线来产生面部图像，与视频摄像头不同，热成像技术并不需要有较好的光源，即使在黑暗的情况下也可以使用。人脸识别技术优点是非接触性的，缺点是要比较高级的摄像头才可有效高速地捕捉面部图像。

指纹识别：指纹识别技术是通过取像设备读取指纹图像，然后用计算机识别软件分析指纹的全局特征和指纹的局部特征，特征点如嵴、谷、终点、分叉点和分歧点等，从指纹中抽取特征值，可以非常可靠地通过指纹来确认一个人的身份。指纹识别的优点表现在研究历史较长，技术相对成熟；指纹图像提取设备小巧；同类产品中，指纹识别的成本较低。其缺点表现在指纹识别是物理接触式的，具有侵犯性；指纹易磨损，手指太干或太湿都不易提取图像。

虹膜识别：虹膜识别技术是利用虹膜终身不变性和差异性的特点来识别身份的，虹膜是眼睛构造的一部分，虹膜中心有一圆形开口，称为瞳孔，犹如相机中可调节大小的光圈，内含色素决定眼睛的颜色。每个虹膜都包含一个独一无二的基于水晶体、细丝、斑点、凹点、皱纹和条纹等特征的结构。虹膜在眼睛的内部，用外科手术很难改变其结构；由于瞳孔随光线的强弱变化，想用伪造的虹膜代替活的虹膜是不可能的。除了白内障等原因外，即使是接受了角膜移植手术，虹膜也不会改变。虹膜识别技术与相应的算法结合后，可以到达十分优异的准确度，即使全人类的虹膜信息都录入到一个数据库中，出现认假和拒假的可能性也相当小。和常用的指纹识别相比，虹膜识别技术操作更简便，检验的精确度也更高。

手形识别：手形指的是手的外部轮廓所构成的几何图形。手形识别技术中，可利用的手形几何信息包括手指不同部位的宽度、手掌宽度和厚度、手指的长度等。经过生物学家大量实验证明，人的手形在一段时期具有稳定性，且两个不同人手形是不同的，即手形作为人的生物特征具有唯一性，手形作为生物特征也具有稳定性，且手形也比较容易采集，故可以利用手形对人的身份进行识别和认证。手形的测量比较容易实现，对图像获取设备的要求较低，手形的处理相对也比较简单，在所有生物特征识别方法中手形认证的速度是最快的。然而手形特征并不具有高度的唯一性，不能用于识别，但是对于一般的认证应用，它足可以满足要求。

掌纹识别：掌纹具有稳定性和唯一性，利用掌纹的线特征、点特征、纹理特征、几何特征

等完全可以确定一个人的身份,因此掌纹识别是基于生物特征身份认证技术的重要内容。目前采用的掌纹图像主要分为脱机掌纹和在线掌纹两大类。脱机掌纹是指在手掌上涂上油墨,在一张白纸上按印,然后通过扫描仪进行扫描而得到的数字化的图像。在线掌纹则是用专用的掌纹采样设备直接获取数字化的图像,图像质量相对比较稳定。随着网络、通信技术的发展,在线身份认证变得更加重要。与指纹识别相比,掌纹识别的可接受程度较高,主要特征较明显,而且提取时不易被噪声干扰。

人耳识别:人耳具有独特的生理特征和观测角度的优势,使人耳识别技术具有相当的理论研究价值和实际应用前景。从生理解剖学上来说,人的外耳分耳廓和外耳道。人耳识别的对象实际上是外耳裸露在外的耳廓,也就是人们习惯上所说的"耳朵"。一套完整的人耳自动识别系统一般包括以下几个过程:人耳图像采集、图像的预处理、人耳图像的边缘检测与分割、特征提取、人耳图像的识别。目前的人耳识别技术是在特定的人耳图像库上实现的,一般通过摄像机或数码相机采集一定数量的人耳图像,建立人耳图像库,动态的人耳图像检测与获取尚未实现。

2. 行为生物特征识别

行为生物特征识别包括步态识别、击键识别、签名识别等。

步态识别:步态是指人们行走时的方式,这是一种复杂的行为特征。步态识别主要提取的特征是人体每个关节的运动。尽管步态不是每个人都不相同的,但是它也提供了充足的信息来识别人的身份。步态识别的输入是一段行走的视频图像序列,因此其数据采集与脸相识别类似,具有非侵犯性和可接受性。但是,由于序列图像的数据量较大,因此步态识别的计算复杂性比较高,处理起来也比较困难。尽管生物力学中对于步态进行了大量的研究工作,基于步态的身份鉴别的研究工作却是刚刚开始。到目前为止,还没有商业化的基于步态的身份鉴别系统。

击键识别:这是基于人击键时的特性(如击键的持续时间、击不同键之间的时间、出错的频率以及力度大小等)而达到进行身份识别目的的一种方法。

签名识别:读者应该都很熟悉在银行的格式表单中签名作为自己身份的标志。将签名数字化是这样一个过程,测量图像本身以及整个签名的动作,包括每个字母以及字母之间的不同的速度、顺序和压力。签名认证的困难在于,数据的动态变化范围大,但事实表明人们的签名在不同的时期和不同的精神状态下是不一样的,这就降低了签名识别系统的可靠性。签名认证按照数据的获取方式可以分为两种,离线(off-line)认证和在线(on-line)认证。离线认证是通过扫描仪获得签名的数字图像;在线认证是利用数字写字板或压敏笔来记录书写签名的过程。离线数据容易获取,但是它没有利用笔划形成过程中的动态特性,因此较在线签名容易被伪造。

生物统计学不是绝对可靠的。首先,它们在非控制环境中很容易伪造,比如,人脸识别系统可能被照相机拍摄的图片,或者甚至可能是素描欺骗。在便利性上也存在一些问题,因为用户生物统计学必须在注册和每次认证时安全获取,这就隐含着一个请求受控制环境的时间消耗,以便确认用户不是因为其他人的利益被强迫进行自我认证。

生物认证机制在非控制的环境中必须谨慎使用,因为存在着滥用机制的可能,可能是攻击者假冒,也可能是强迫合法用户认证来获得访问系统的合法权限。

2.1.3 基于口令的认证

加密认证依赖于对实体拥有的密钥的认证。通常在密钥空间中随机选择密钥，密钥的长度取决于所用的算法和所需的安全等级，可以从 100b 左右到 1000b。经验表明人们很难记忆随机形式的秘密，甚至是七八个字符的密钥。但是用所有大写和小写字母加上 0 到 9 的数字构成的一个随机的 8 字符的密钥，其随机性不超过 48b。

加密密钥经常存储在计算机的安全存储器中，或者使用特殊的设备来存储密钥，如抗篡改加密服务器或智能卡。但是，这些方法要么不太方便要么成本太高，不是所有的设备都能抵抗篡改，公钥需要的内存也很缺乏。因此，有必要建立仅依赖于人能记忆的短秘密的安全通信。

1. 两种形式

口令是目前应用最广泛的认证方法，它有以下两种形式。

(1) 文化的。文化法基于概念或词组的存储，有时也称为认知或语义口令。

文化口令(cultural passwords)基于推理过程产生请求口令。与随机口令(任何单词或短语都可以)相比，文化口令在一些方面更有意义。文化口令意味着用户有一个挑战问题，响应问题获得答案(如图 2-2 所示)。这个理论中用户几乎不存在忘记口令的问题，因为文化口令要求的回忆是建立的事实或选项。



图 2-2 文化口令

(2) 随机的。随机法使用一个随机的符号或数字序列(它们可能是由每个用户随机产生或选择的)，如果是一个单词就称为口令，如果仅由数字组成就称为 PIN，如果由多个单词组成就称为通行码(pass phrase)。

目前，大部分的认证机制都使用随机口令，且越长越好。但现实问题是口令常常需要用户记忆在脑子里，因为记录到某个地方既不安全，使用起来也不方便。可是越是随机的口令越不容易记住，同样口令越长，往往记忆起来也越困难，而且如果一个口令因为记忆的错误所造成的损失可能会比口令被窃取所带来的损失更大，这是安全性和实用性之间的矛盾，如果解决不好这对矛盾，系统中就难免有安全漏洞存在。

大家都知道选取一个好的口令非常重要，但下面一些不好的做法却普遍存在。

① 使用个人特征信息(或部分信息)作为口令或部分口令，如口令中使用自己或亲属的生日信息或几个人生日信息的结合，或者使用自己的居住信息，包括街道号、门牌号和楼层数等；

② 多种用途使用同一个口令，如不同银行卡使用同一个口令，不同邮件信箱使用相同口令等；

③ 口令长期不更换。

如何才能产生容易记忆但又不容易被猜测到的口令呢？如果需要一个计算机账户所使用的口令，则可以选择一个容易记忆、自己喜欢的“只言片语”来生成。下面举几个简单例子说明(见表 2-1)。

表 2-1　口令的选择举例

喜欢的“只言片语”	生成的口令	喜欢的“只言片语”	生成的口令
大家星期六不上班	Dj * 76bsb	我怎么去见我的妈	Wzmq95dm
星星之火可以燎原	** zhk1ly	无论如何就是不行	5lrh94bx
我的心里七上八下	Wdxl7s8x	哎哟算了吧算了吧	A1328slb

当然有些“只言片语”可以生成多种不同的口令，只要选取自己喜欢的一种就行了。使用这种方法可以生成容易记忆且不易被猜到的口令。

2. 三个种类

自 20 世纪 90 年代初以来，口令认证密钥交换协议的设计与分析成为研究热点，主要分为以下三类。

1）两方口令认证密钥交换协议（2PAKE）

1992 年，Bellovin 和 Merrit 首次提出了一个两方口令认证及 Diffie-Hellman 密钥交换协议，即加密密钥交换协议（EKE）。两方口令认证密钥交换协议（2PAKE）通常使用“用户-服务器”模型，为了相互认证并建立会话密钥，用户和服务器需要事先共享一个口令，对于有 n 个用户的系统来说，若每两个用户都共享一个口令，则每个用户需要记住 n 个口令，整个系统需要存储 $O(n^2)$ 个口令，从而限制了协议在实际中的应用。为了解决这个问题，出现了许多 Client-to-Client 口令认证密钥交换协议（C2C-PAKE）。

2）Client-to-Client 口令认证密钥交换协议

包括基于口令认证的三方密钥交换协议（3PAKE）和跨域口令认证密钥交换协议（C2C-PAKE）。

基于口令认证的三方密钥交换协议（3PAKE）中的用户只需要和一个可信服务器 S 共享一个简单的口令（或者与口令相对应的验证信息），通过 S 的协助，通信双方完成相互认证和会话密钥的交换。

跨域口令认证密钥交换协议（cross-realm C2C-PAKE）可实现位于不同区域且持有不同口令的两个用户，通过各自域服务器协助协商出共享的会话密钥，建立起一个安全的通讯信道的过程。2002 年，Byun 等首次提出了一种跨域环境中基于口令认证的密钥交换协议。

3）群组口令认证密钥交换协议

群组口令认证密钥交换协议（GPAKE）可实现在开放性网络中，群组成员通过认证的群组密钥交换协议产生共享密钥，并利用该密钥来保证组播消息的保密性和完整性。因此群组密钥交换协议对于安全的视频或电话会议等应用环境非常重要。

安全的口令认证密钥交换协议应该具备以下安全属性。

（1）双向认证（mutual authentication）。双向认证要求通信双方能够验证对方的身份。

（2）会话密钥安全（session key security）。会话密钥安全是指攻击者在不知道口令的情况下，不能恢复出利用该口令建立的会话密钥。

（3）抵抗离线字典攻击（off-line guessing attack resilience）。为了便于记忆，会经常使用低熵的口令，但这种口令本质上容易遭受字典攻击。把所有用户可能使用的口令组成的集合叫做口令字典。抵御离线字典攻击是指不存在如下成功的攻击者：攻击者拦截并存储协议执行时的消息，在离线的状态下，攻击者可以选择字典中的一个候选口令，并利用拦截

的消息验证该候选的口令是否为正确的口令，如果不正确则选择另一个候选口令直到猜测正确为止。

(4) 抵抗在线字典攻击(on-line guessing attack resilience)。主要有两种方式：一种是抵御不可检测的在线字典攻击，是指不存在如下成功的攻击者——试图在在线的状态下验证猜测的口令，而失败的猜测不被服务器检测到；另外一种是抵抗计算机程序的在线字典攻击，是指不存在如下成功的攻击——在在线的状态下，程序自动加载，验证猜测的口令，而失败的猜测不被服务器检测到。

(5) 完善前向安全(perfect forward secrecy)。长期密钥的泄露不会威胁以前会话密钥的安全性，即以前的通信不会因长期密钥的泄露而受到威胁。

(6) 已知会话密钥安全(known session key security)。通信双方每执行一次密钥协商协议，会生成一个唯一的会话密钥，如果会话密钥的泄露不会导致被动攻击者得到其他的会话密钥或主动攻击者假冒协议参与者，则称协议具有会话密钥的安全性。

(7) 抵抗服务器遭侵害的假冒攻击(verifier-compromise impersonation attack resilience)。抵御服务器遭侵害的假冒攻击是指不存在如下成功的攻击者：有两种攻击形式，一是假设攻击者盗取了服务器中 A 的验证信息，在不实施字典攻击的情况下，攻击者能够假冒其他用户(发起者)与 A(响应者)通信；二是假设攻击者盗取了服务器中 A 的验证信息，在不实施字典攻击的情况下，攻击者能够假冒 A(发起者)与其他用户(响应者)进行通信。

(8) 抵抗口令泄露假冒攻击(password-compromise impersonation attack resilience)。抵抗口令泄露假冒攻击是指不存在如下成功的攻击者：如果用户 A 的口令 pw_A 泄露，那么窃取 pw_A 的攻击者自然能够假冒 A 与其他用户通信，从而假冒其他用户与 A 建立共享会话密钥。

下面将分析应用 Diffie-Hellman 密钥交换的加密密钥交换协议(EKE)(见表 2-2)。虽然原始的协议有许多潜在的缺陷并缺乏安全证明，但有助于理解基于口令的认证。

表 2-2 基于口令协议用到的表示符号

π	A 拥有的短密钥，如口令
p	大素数(至少为 1024b)
q	素数(160b)，且 $q \mid p-1$
G	Z_p^* 的子群，G 常是阶为 q 的子群，有时等于 Z_p^*
g	G 生成元
r_A, r_B	A 和 B 分别选择的随机整数，与 G 的阶的长度相同
t_A, t_B	短暂公钥：$t_A = g^{r_A}$，$t_B = g^{r_B}$，在 Z_p 内计算
x_A, x_B	A 和 B 的长期私钥
y_A, y_B	A 和 B 的公钥：$y_A = g^{x_A}$，$y_B = g^{x_B}$，这些公钥有某些标准方法进行书面证明，这里不考虑
Z_{AB}	共享秘密
K_{AB}	生成的会话密钥

EKE 的大致思想是传输用口令加密的临时公钥作为共享密钥。只有知道口令的实体才能完成协议。这个思想适用于不同公钥策略的临时密钥。这里只考虑 Diffie-Hellman 密钥交换，两个实体都选择临时公钥，口令用于加密临时公钥，如图 2-3 所示。

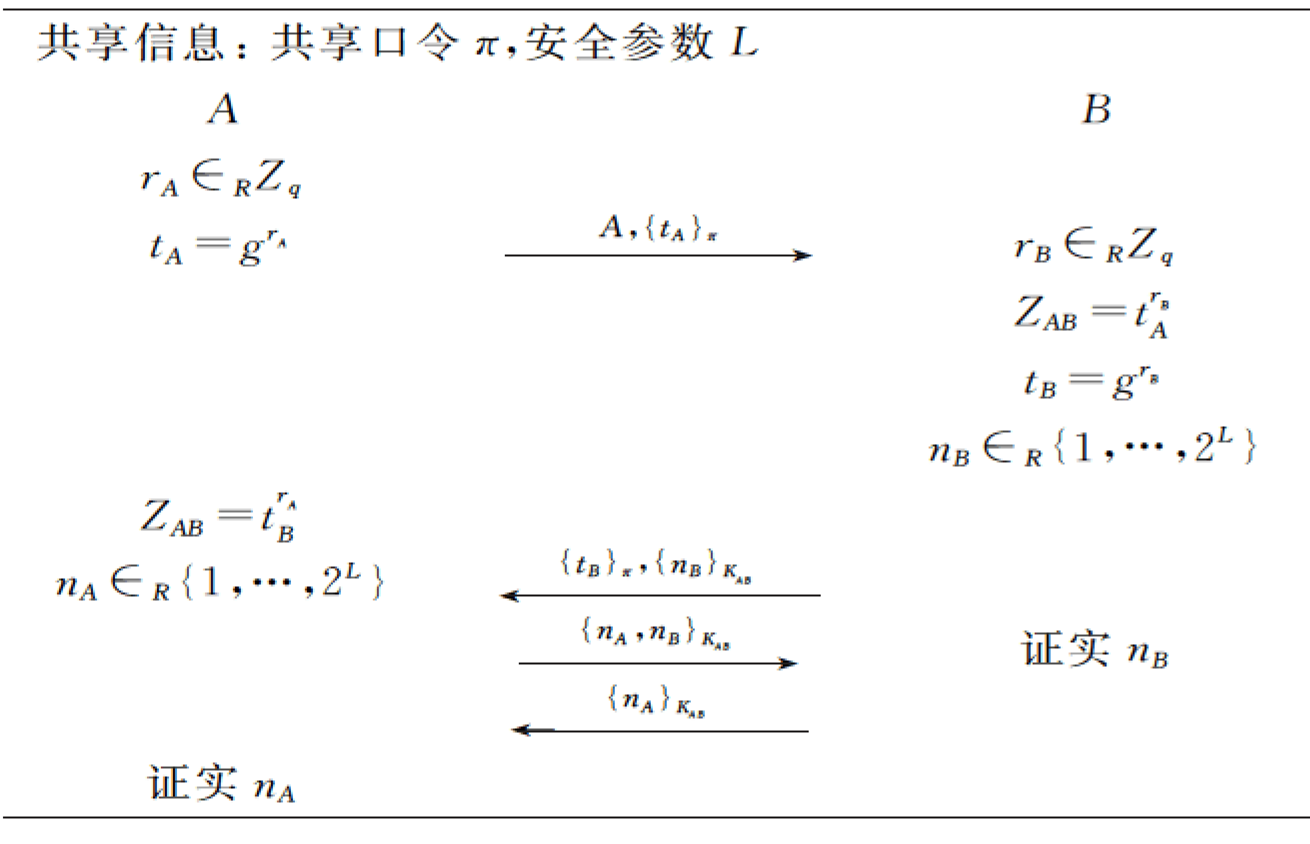


图 2-3　基于 Diffie-Hellman 的 EKE 协议

在基本 Diffie-Hellman 密钥协商中，虽然由共享秘密 Z_{AB} 产生会话密钥 K_{AB} 的密钥派生函数没有具体说明，仍有 $Z_{AB} = g^{r_A r_B}$。图 2-3 中双方都需要进行两次幂运算，这和普通 Diffie-Hellman 是一样的。

Bellovin 和 Merritt 引入对 EKE 的分割攻击(partition attack)的思想，攻击者猜测口令试图解密 $\{t_A\}_\pi$ 和 $\{t_B\}_\pi$，并检验明文结果是否是一个有效的 Diffie-Hellman 临时值。如果不是，则猜测的口令是错误的，并丢弃之。经过多次的协议运行后，就可以分割出有效的口令集和无效的口令集。

2.1.4　基于验证码的认证

当前，有许多专门的破解工具可以在线暴力破解口令(如 BBS 口令、在线邮件口令等)，给口令认证造成了极大的安全威胁。逆向图灵测试(Reverse Turing Test，RTT)是一个区分人与计算机程序的方法，因为 RTT 很容易被人识别，但是对于自动程序来说很难识别。它很有希望击败有自动程序加载的在线字典攻击。一个典型的 RTT 就是反常图形化的验证码，如图 2-4 所示。因此，目前在使用用户名和口令进行注册和登录之前，网络服务提供者要求用户通过 RTT，即在页面上按要求输入验证码后才能正常进入账号，这样可以有效地抵御穷举攻击和字典攻击。由于每次页面访问的验证码都不相同，同时安全程度较高的验证码使得程序化的信息提取变得不可能，而必须由用户进行识别输入。由于人工因素的引入，使得原本单位时间内高密度的攻击骤减，由基于计算机计算能力的高频攻击转化为基于人工输入的低频攻击，针对简单口令的穷举攻击和字典攻击都将会耗费大量的时间和人力，从而导致口令遍历猜测的攻击方式失效。

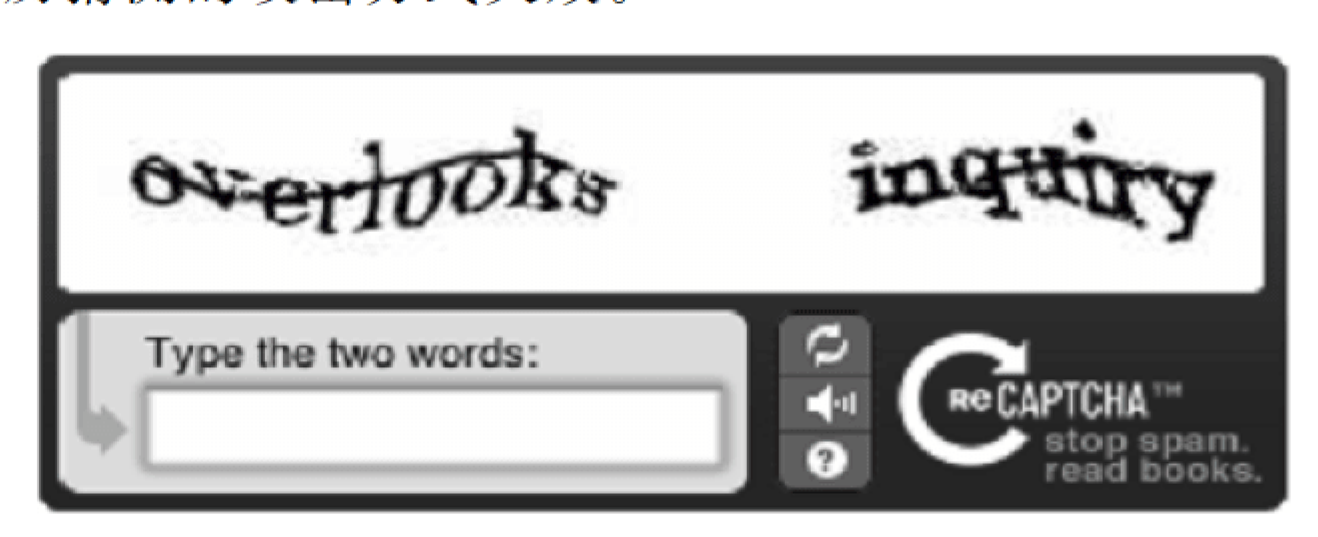


图 2-4　反常图形化的验证码

其他一些 RTT 是基于语音识别的方法。它将一些单词的声音片段和一些随机选择的混扰杂音放在一起输出,使用者需识别其中声音的内容。这种系统是利用人和计算机在语音识别方面的差异实现的。对于一些视觉存在问题的人群来说,以语音作为 RTT 系统是十分有用的。

具体的说,图形化的验证码就是这样一类技术。服务器生成随机字符串 R,经变换成 $\varphi(R)$后发送给用户,由于机器不能识别出变形图片 $\varphi(R)$中的字符串 R,只有人才能够识别,这保证了客户端参与者必须是人,从而避免了攻击者利用机器进行自动的在线字典攻击。通俗地说,就是由服务器随机产生文本序列,然后与背景图片进行信息融合生成最终的验证码。图形化的验证码有如下特点:验证码是随机生成的;验证码只是为防止程序猜测等目的而生成的无意义字符串,不需要用户记忆;验证码只对当前服务有效,并具备时效性,只在一定时间内有效(如一般网页规定验证码的有效时间为 5 分钟)。

验证码作为一种安全技术,图形化的验证码的安全强度主要基于图像识别的难度,其具备一定的信息隐藏性,使得一般程序化手段难以进行提取,因此,提高验证码的安全性,必须从增加信息提取难度入手。

图形化的验证码从两个方面增加了信息提取难度:一方面,在信息传输和页面显示中不存在直接可提取的验证码文本,要进行图像-文本的程序转换必须通过图像识别;另一方面,针对图像识别技术,可在信息融合过程中添加干扰信息,同时进行图像混杂、扭曲或变形处理,增加图像识别的难度,从而提高图像识别的算法复杂度,降低识别正确率,以达到用户可识别,而无法进行程序化识别的最终目标。

提高图形化验证码安全强度的方法主要有以下三种。

(1) 背景干扰:选择变化的背景图片,是降低图像识别率的基本方法。通常有不同的背景色、背景点、背景图、渐变背景色、网格背景等。

(2) 前景变形:通过对前景图形(通常是数字、字符、汉字等)进行倾斜、膨化、波浪化、风化、背景图形凹凸、球状扭曲等效果来形成。

(3) 信息码变换:通过改变字符、数字等的字体、颜色、位置、大小和增加多余线条来干扰图像的识别。

2.1.5 基于图形的认证

基于文本的口令认证以其简单易行、使用面广的特点成为现在使用最为广泛的身份认证技术。在计算机系统中,操作系统、网络、数据库均采用了口令验证的形式。但是这种方式存在着较大的安全隐患,主要是可记忆性与抗破解能力之间的平衡。如果设置易于记忆的口令,那么攻击者可以通过搜索口令字典数据库的方法对其进行字典攻击;如果设置的口令长度过短,则口令空间过于狭小,攻击者可以采用暴力破解的方法在有效的时间内找到口令。要提高抗破解能力的唯一方法是增加口令的长度,并少用人类熟悉的字符串,但是这样可记忆性就会减弱。其次,文本口令的口令空间只包含 94 个字符,这直接限制了其安全性的上界。

图形口令作为文本口令的替代者,很好地解决了文本口令面临的字典攻击、口令难于记忆、口令管理困难等问题。图形口令是通过让用户在图形用户界面上显示的图像中按照特定的顺序进行选择来工作的,是利用人类对图形记忆要优于对文本记忆的特点设计出来的

一种新型口令。用户不用记忆冗长的字符串而是通过识别或记住图形来进行身份认证。图形口令提供比文本口令更强的安全性。

图形口令还可以解决文本口令可能经常更换的问题。生物学家的研究表明了这样的一个事实：人类记忆图形的能力会比记忆文本的能力强的多。此外图形口令的口令空间会比文本口令的口令空间大的多，更容易让口令抵御诸如字典破解之类的攻击。正是因为上面的这些优点，人们对图形口令的兴趣才日益高涨。

从技术上，图形口令可以分为两类：基于识别型和基于回忆型。

(1) 基于识别的系统。该系统要求用户在许多分散注意力的图像中选择目标图像，这个方法基于纯视觉记忆，利用识别先前看到的可视客体的能力。

一种基于识别型的图形口令身份验证要求用户记忆预先选定的一些特定图片，在验证阶段系统从图案库中随机产生一组图片，让用户从中间选择预先设定的图片，从而实现身份验证的过程。

PassfacesTM 是由 RealUser 公司开发的一个身份认证系统，目前主要应用于 PDA 上。它基于人类识别人脸比识别其他图形都更容易的理论，利用人脸图片作为认证媒介。用户在设定口令阶段，从人脸数据库挑选出 4 幅图像作为口令。验证阶段，用户看到一个由九个人脸组成的 3×3 网格，包括一个口令图像和 8 个迷惑图像。认证时，用户在屏幕点击自己事先指定的人脸图片，该过程重复四次，用户全部选对预先设定的人脸图像就可以通过验证。因为人们对人脸记忆更容易，识别更迅速，因而缩短了验证时间。PassfacesTM 使用方便，但是口令空间小，安全性不高。另外，一些学者指出，这个系统并不适用于患有面孔不可识别症的人，因为他们不能识别不同的人脸。由于人有人种、性别、年龄等的区分，用户选择人脸时有一定倾向性，更加降低了 PassfacesTM 的安全性。

“肩窥”指使用直接的观察来获取信息，就如从某人的肩膀上方来查看。在拥挤的公共场所，肩窥是窃取信息的一种有效方法，当别人填表、在 ATM 机输入 PIN 码或者在公共付费电话使用电话卡时，其私密信息容易被他人获取。“肩窥”也可以使用双眼望远镜或者其他视觉增强设备来远距离实现。在使用传统文本口令的情况下，用户只能依靠身体遮挡工作区来避免肩窥。在采用图形口令方法后，即使偷窥者看到用户输入过程，但仍然无法确定用户设定的图形，并且在每次验证时展现的图像阵列都不相同，所以能够有效防止肩窥。例如一种图形和字母-数字口令相结合的模型可以有效地防止肩窥。该系统提供许多不同的小图片，每个图片都有八种微小变化(如图 2-5 所示)，人类可以识别，计算机识别程序则难以识别。

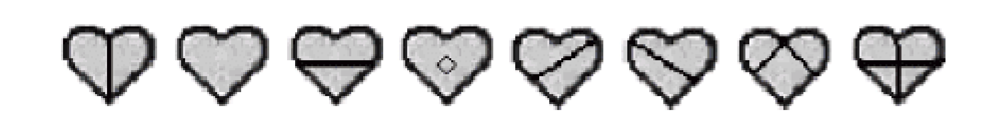


图 2-5　有微小变化的图片

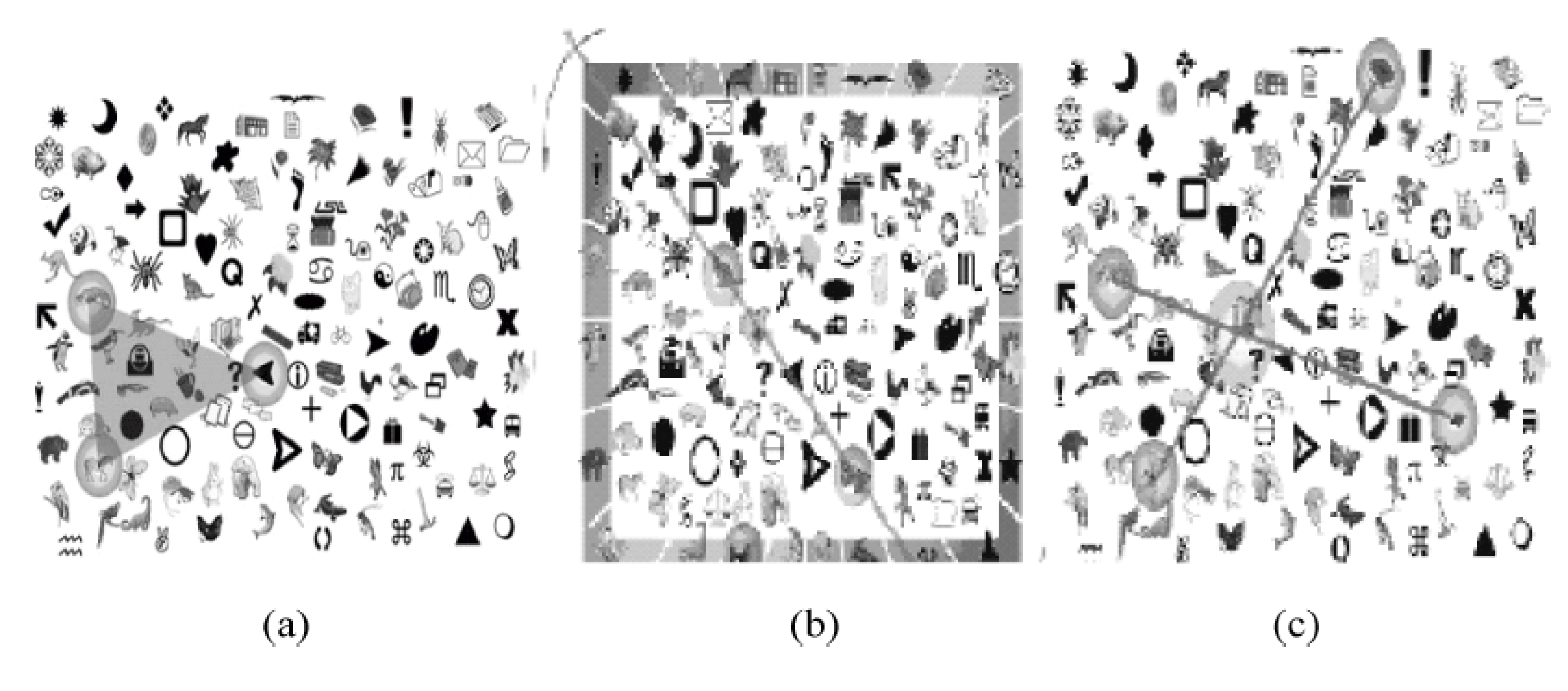
用户初次使用这种系统的时候，首先需要从图片当中选择四种图形，然后对这四种图形的每种变化指定一个代表的字符串，不同的用户为相同的图片指定的字符串可以是不同的。

在认证过程中，系统会随机生成 11×11 个图形，其中有 4 个是用户事先指定的图形，而它们以何种变化出现是随机的。用户需要识别出这 4 个图片，然后在系统输入框中依次输入指定的字符串。

这种模型的认证过程复杂且耗时,但是它的安全性非常高。首先,它的口令空间很大。其次,它将图形和文字口令结合起来,选择相同图形的用户指定的文字口令不一定相同,因此每个用户的口令相对是独立的。最后,由于识别图形的过程,是在用户的大脑当中进行的,并不对图形做任何操作,即使对这一过程进行监视,也无法知道用户指定的图形;由于每次出现的图形都不一定相同,用户输入的口令也不一定相同,即使被如键盘扫描、偷录等方法记录了某一次输入,也无法分析出用户的所有口令。

另外一种实现方式是由用户预先选择一些图形物体,在验证的时候系统显示出一个由许多图形组成的阵列,用户需要识别出这些物体,并且移动一个固定的框架使阵列中预先定义的物体全部落在框架中,通过多次重复此过程来防止随机选中的可能。这种方法可以有效地防止"肩窥"。

例如(如图 2-6(a)所示)由用户目测出指定图形构成的三角形,在该虚构三角形当中,用鼠标单击以通过认证。另外两个类似的例子如下。一个是通过旋转外框使指定图形和另外两个内框当中存在的指定图形成直线进行认证(如图 2-6(b)所示),另一个则是由用户单击四个指定图形构成的两条虚拟直线的交叉点以通过身份认证(如图 2-6(c)所示)。



(a) (b) (c)

图 2-6 防止"肩窥"的图形口令

(2) 基于回忆的系统。该系统要求用户在网格中画出先前画的客体,或者在单个图像中识别目标客体。这个方法基于视空间记忆的视觉和空间两个方面,以及精确的动作。

一种基于回忆型的图形口令身份验证则是要求用户重复以前设定的一个过程。DAS(Draw A Secret)是目前较广泛应用于 PDA 上的一种身份认证的系统。用户初次使用系统时,要求用户在 5×5 的 2D 栅格上画出口令,系统按照一套本身能够识别的格式记录口令。在验证阶段,系统显示同样的栅格要求用户重复原来的设定过程,如果用户画出的图形按照以前设定的顺序经过相同的方格则通过验证。

人机互动系统指用户通过在虚拟环境中重复某一过程以通过认证的系统。例如 Passlogix 系统提供一些 2D 场景,如厨房等。用户与场景进行互动,如煮饭,喝酒等。该认证过程完全看用户的选择,弹性非常大。比如用户首先按照顺序选择了一个闹钟、手表、钱包,注意一定要点中要选的物品,在下次认证的时候如果按照顺序同样选择了那个闹钟、手表、钱包,就可以通过认证。

随着虚拟现实技术的发展,目前已经出现了一些概念性的 3D 人机互动系统。它们的系统界面很真实,用户可以进行虚拟漫游。系统识别用户操作的物体坐标、行为等,确定是否认证成功。这样的系统提供互动操作更多,并且可以结合现实中的物品,如借助读卡器在

系统当中要求用户以磁卡进行操作或借助手写板要求用户在系统中进行手写识别等，从而大大提高了系统的安全性。只是就目前来看，这样的系统对硬件系统要求比较高，目前还处在实验阶段，主要应用方向是对安全性需求非常高的系统。这种系统要真正广泛应用，还需要假以时日。

图形口令与传统文本口令相比较在安全性上具有很大优势。可以通过增大图案库的容量来扩大口令空间，提高系统的安全性的同时也不会降低可记忆性。对于这种新型的口令，很难采用现有的攻击方法来攻击。由于图片库大，使用暴力破解是不可行的，而传统的字符只有 94 个（包括空格），其口令空间受到限制。从攻击者的角度看，攻击者必须了解并精确复制系统图库，难度加大。图形口令采用鼠标输入，比传统口令的键盘输入更加难以猜测。攻击者使用间谍软件来跟踪键盘输入容易，但是跟踪鼠标输入困难，并且由于用户输入图形操作和用户当前所使用的图形窗口位置、大小以及时间信息都有关，盗取口令更加困难。从保管口令的角度看，图形中包含的信息庞大且不容易用语言描述，不容易泄露出去。

2.1.6　基于令牌的认证

认证可以基于客户持有的某物，这样的一个客体一般称为令牌。

2008 年初，中国银行（http：//www. boc. cn）开发了全国统一的新版网上银行系统（https：//ebs. boc. cn/BocnetClient/LoginFrame. do? _locale＝zh_CN）。在安全保障上，新版网银首次大规模应用了国际流行的安全认证工具——动态口令牌（E-TOKEN），客户在登录以及重要交易操作中均需输入动态口令进行验证。

该动态口令牌是一种内置电源、密码生成芯片和显示屏、根据专门的算法每隔一定时间（如 60 秒）自动更新动态口令的专用硬件，如图 2-7 所示。基于该动态密码技术的系统又称一次一密（OTP）系统。该装置拥有内置芯片和一个可以显示多达 6 位数字的 LCD 窗口，但其体积很小，可以系在钥匙环上。在柜台发售这种装置时，与网银用户绑定建立一对一对应关系，使用唯一的 128 位种子将其初始化；其内部芯片每分钟都会使用一种算法，组合该种子与当前时间，生成一个随机的数字。而在网银认证服务器则采取和这个动态密码器同一种算法产生该随机数字，保证动态密码器和网银服务器的单一认证。



图 2-7　中国银行动态口令牌

另一个例子是串行总线 USB（Universal Serial Bus）认证令牌，它包含一个智能卡芯片和一个智能卡读卡器。这些设备一般包含一个私钥，一个公钥和一个认证机关颁发的证书。远程系统发一个挑战给令牌来验证用户确实拥有相应的私钥；然后，系统询问数据库验证证书上的名字是否与授权访问的身份一致。

令牌既可以由硬件设备也可以由软件设备提供。硬件令牌有一个终生电池，并将安全码显示于 LCD 显示屏上。软件令牌运行于用户的私人机器上，使用用户存储于台式机上的而非硬件令牌上的密钥。

尽管这样的硬件设备比普通口令更能提供安全利益，但是它们也有一些缺陷。设备有相关的花费，这使得它们不像其他机制那样可行。用户还必须携带硬件令牌。如果用户有许多应用要求使用硬件令牌，那么携带所有这些令牌就很繁重了。此外，用户需要某种方法来记住用于不同系统的令牌。软件令牌通过存储密钥在自己的计算机上减轻了这些问题，

但是访问特殊系统时，这会将用户与机器绑定起来：软件令牌没有硬件令牌安全，因为用户的计算机可能遭到攻击。

最后，一些基于令牌的系统可能要求用户记住 PIN，这将所有的问题都与无线索的回忆联系起来了。

2.1.7 双因子认证

登录网络的用户需要通过用户名和密码来进行身份验证，像电子信箱、网上银行等大多是静态密码。但静态密码有很大的安全隐患，如盗号木马可以在用户登录时获取密码，利用黑客工具破解静态密码也较为简单。最常用的解决策略是双因子认证，因为它将两个系统联系起来了，允许每个系统弥补其他系统的不足。

双因子认证是密码学的一个概念，是在静态密码的基础上，使用了认证增强机制。认证机制中包含有两个认证因子即被称作双因子认证。认证因子包含有个人识别码、相应记号、生物学特征等，人们熟知的手写签名和键盘输入密码均在其中。现在，一般常见的双因子认证一般结合密码以及用户手中的口令卡、IC 卡、USB Key、动态口令牌等进一步验证用户的身份，从而抵御非法访问者，提高认证的可靠性。比如，基于令牌的认证系统一般与口令相联系。这阻止了用户因为令牌丢失(因为攻击者不知道口令)或口令被盗(因为攻击者没有令牌)的风险。

国内外的银行、民航等行业已经逐步发展了双因子认证技术。经常网上购物的用户所熟悉的口令卡与密码结合的银行认证方式就是一个很好的例子：在网上划款时，用户不仅需要输入设置的密码，还要根据网页的提示，从口令卡中找到正确的密钥并输入(每次密钥都不相同，属性等级较高导致安全性提高)，两种认证都通过才能操作自己的账户。图 2-8 给出的是工商银行电子银行口令卡。



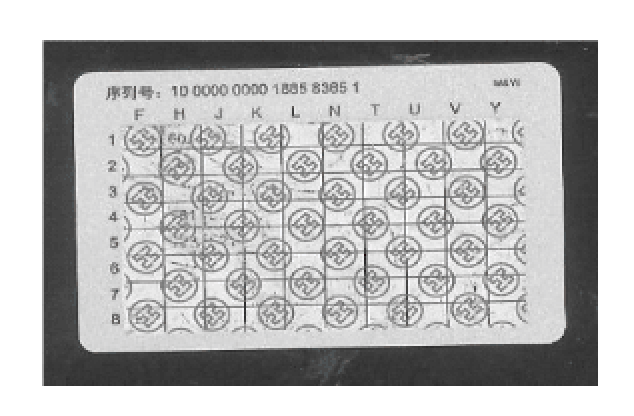


图 2-8 工商银行电子银行口令卡

对一些攻击而言，双因子认证几乎没有增加安全性。比如，ATM 机器使用双因子认证：要求银行存款人拥有 ATM 卡，并知道取款的口令。但是，这些双因子没有提供给存款人足够的安全，存款人可能被绑架，被强迫拿出 ATM 卡并泄漏口令。

基于硬件信息的认证方式是最近发展起来的，它是通过计算机本身的唯一硬件特征来标识使用者的身份，结合 PIN 码的使用，可以实现一种高强度的双因子认证。这种认证依然是建立在公钥密码体制之上的。它的基本假定就是对于固定的用户，其使用的计算机也是相对固定的(在公用的计算机上是不应该执行任何涉及个人机密的操作的，否则安全得不到任何保障)，那么，通过对这台计算机的识别，加上对当时使用计算机的用户识别，就可以实现对用户的远程认证。其难点和重点在于识别计算机的唯一硬件特征，比如网卡都有一个全球唯一的 MAC 地址，网卡生产商都遵循统一的规定，按照统一的分配来给自己生产的

网卡指定 MAC 地址。同样地，对于 CPU、硬盘、主板等其他计算机部件，都存在着相应的协议和规范。而这些参数的联合，足够构成一个全球唯一的硬件标识号码。其工作原理是：首先对合法用户的计算机进行硬件特征采集，通过对于硬件标识号码的实时获取，可以实时地认证一台具有唯一特征值的计算机是否为已经注册的合法使用者。而认证的另外一个过程发生在认证开始之前，用户要启动认证过程，首先要输入自己的 PIN 码。这样，通过对用户口令和计算机硬件特征值的联合识别，可以以相当高的安全性来确认用户的身份；而且由于没有作废列表查询更新等问题，认证是相当快速的。这样的认证系统与现有的业务和网络系统的整合也相对简单，它本身是相对独立的系统，其实现也并不复杂，非常容易整合到现有的网络体系之中。

2.2　访问控制

访问控制（access control）是实现既定安全策略的系统安全技术，目标是防止对任何资源（如计算机资源、通信资源或信息资源）进行非授权的访问。访问控制系统一般包括主体（subject，发出访问操作、存取要求的主动方，通常指用户或用户的某个进程）、客体（object，主体试图访问的一些资源）、安全访问策略（一套规则，用以确定一个主体是否对客体拥有访问能力）。

2.2.1　访问控制矩阵

这里定义主体为系统的一个用户，客体为系统资源。两个基本的概念是访问控制列表（ACLs）以及能力表（C-lists），ACLs 和 C-lists 都来自于 Lampson 的访问控制矩阵，即每个主体占一行，每个客体占一列。S 行与 O 列的交接处表示主体 S 允许访问客体 O。表 2-3 表示的是一个访问控制的例子，这里 x、r 以及 w 依次表示执行权、读权和写权。

表 2-3　访问控制矩阵

	操作系统	记账程序	核算资料	保险数据	工资单数据
Bob	rx	rx	r	—	—
Alice	rx	rx	r	rw	rw
Sam	rwx	rwx	r	rw	rw
Acct. program	rx	rx	rw	rw	r

注意，表 2-3 中记账程序既可以作为主体也可作为客体。通过这种方法加强限制，使得核算资料仅能够被记账程序修改。这里的目的是使得破坏核算资料更加困难。然而，这样并不能防止所有可能的攻击，因为系统管理员 Sam 能够将记账程序替换为一个坏版本而绕过这个保护。但是 Alice 和 Bob 能够访问核算资料发现攻击，虽然系统不允许他们进行修改（不管是故意还是无意）。

1. ACLs 和 C-lists

因为所有的主体和客体都在访问控制矩阵中，它包括所有基于认证决策的相关信息。不过，在管理大的访问控制矩阵时存在一个实际问题。实际中一个系统可能有上百个主体

和上千个客体，在某些情况下，有上万个条目的访问控制矩阵中，当主体对客体进行操作时需要先进行询问，处理这类大型矩阵会给任何可信系统带来不可接受的负担。

访问控制矩阵被分成若干可管理的片段。有两种可行的方法可以分段访问控制矩阵。首先，可以将矩阵存储为若干列，每一列存储与之对应的客体。然后，当一个客体被访问时，它的访问控制矩阵的列会被询问该操作是否被允许。这些列就是访问控制列表(ACLs)。举一个例子，表 2-3 中保险数据的 ACL 可表示为：(Bob，—)，(Alice，rw)，(Sam，rw)，(Acct. program，rw)。

也可以将访问控制矩阵存储为行的形式，每行存储它对应的主体。然后，当一个主体要实行一个操作时，可以询问它的访问控制矩阵，看这个操作是否允许。这个方法就是能力表(C-lists)。举一个例子，Alice 在表 2-3 中的 C-list 为：(操作系统，rx)，(记账程序，rx)，(核算资料，r)，(保险数据，rw)，(工资单数据，rw)。

2. 特权和解决责任混淆

"特权与解决责任混淆"是一个传统的安全难题，在这个难题中，有两个系统资源，一个编译器以及一个名为 BILL 的文件(包含重要的账单信息)，还有一个用户 Alice。编译器能对任何文件进行写操作，包括 BILL，当 Alice 请求编译时，她能够提供文件名(调试信息可以被写入该文件)。但是 Alice 不允许写入文件 BILL，因为她可能破坏账单信息。表 2-4 给出了这种情况下的访问控制矩阵。

表 2-4 特权和解决责任混淆的访问控制矩阵

	编 译 器	BILL
Alice	x	—
编译器	rx	rw

假设 Alice 请求编译，她提供 BILL 作为调试文件名。Alice 没有访问文件 BILL 的特权，所以这个命令失败。尽管如此，编译器代表 Alice 的利益，它有对 BILL 进行重写的特权。Alice 的命令的结果可能毁坏 BILL 文件，如图 2-9 所示。

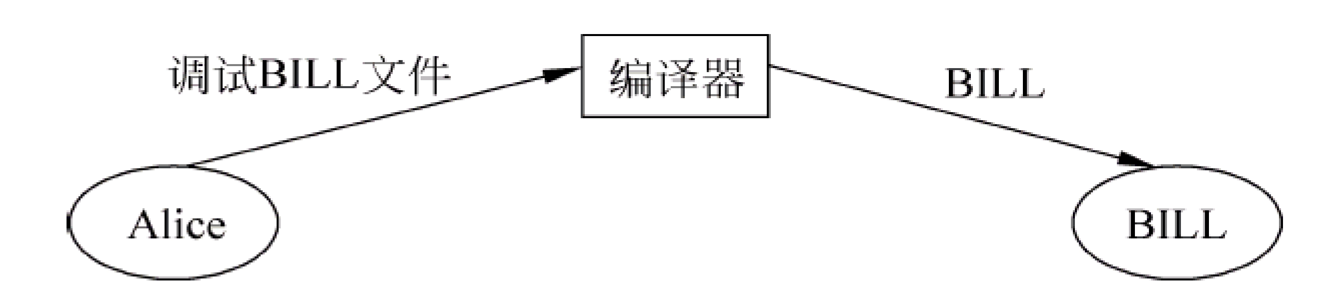


图 2-9 特权和解决责任混淆

为什么这个问题称之为特权和解决责任混淆？编译器代表 Alice 的利益，所以它是代理。编译器是混淆的，因为它应该基于 Alice 的特权进行行动，但却是基于它自己的特权进行行动。

在 ACLs 中很难避免这种混淆，但是 C-lists 却相对很容易避免这个问题。这是因为 C-lists 很容易被指派，而 ACLs 不能。在 C-lists 系统中，当 Alice 请求编译时她很容易就可以把她的 C-list 给编译器。在创建调试文件时，编译器首先检查特权，然后询问 Alice 的 C-list，因为 Alice 没有对 BILL 进行重写的权利，这样图 2-9 的情形就可以避免。

将 ACLs 与 C-lists 的相关优点比较是有必要的。当用户管理自己的文件且想保护数据时，选择 ACLs 比较好。对于 ACLs 而言，改变对某个资源的权利也是很容易的。另一方面，C-lists 很容易进行指派，而且添加、删除用户也很容易。由于具有指派功能，使用 C-lists 很容易避免特权和解决责任混淆。然而，C-lists 的执行更为复杂，开销也更高。因此，在实

际中更多的是使用 ACLs 而不是 C-lists。

2.2.2 访问控制模型

传统的访问控制模型有自主访问控制(Discretionary Access Control,DAC)、强制访问控制(Mandatory Access Control,MAC)和基于角色的访问控制(Role-Based Access Control,RBAC)。DAC 根据主体的身份和授权来决定访问模式,但信息在移动过程中主体可能会将访问权限传递给其他人,使访问权限关系发生改变;MAC 根据主体和客体的安全级别标记来决定访问模式,实现信息的单向流动,但它过于强调保密性,对系统的授权管理不便,不够灵活。总之,DAC 限制太弱,MAC 限制太强,且二者的工作量较大,不便管理。RBAC 则可以折中以上问题,角色控制相对独立,根据具体的系统需求可以使某些角色接近 DAC,某些角色接近 MAC。

1. 自主访问控制

DAC 最早出现在 20 世纪 70 年代初期的分时系统中,是多用户环境下的一种访问控制方式。DAC 自主访问控制根据访问请求者的身份,以及规定谁能(或不能)对什么资源进行什么操作的访问规则来进行访问控制。在 DAC 系统中,访问权限的授予可以进行传递,即主体可以自主地将其拥有的对客体的访问权限(全部或部分)授予其他主体。

访问矩阵模型是一种自主访问控制的典型模型和实现框架。访问控制模型采用访问矩阵来描述系统在某时刻的授权状态,因此矩阵实质是防护机制的抽象表示。在访问矩阵模型中,访问主体能够访问哪些客体,怎样访问这些客体都在访问矩阵中定义。访问矩阵同时又是系统的安全状态。访问矩阵可以用一个三元组(S,O,P)表示,其中,S 表示实施权限的主体集;O 表示客体集,主体在其上实施权限;P 表示访问矩阵。访问矩阵是以主体为行索引、以客体为列索引的矩阵,矩阵中的每一个元素表示一组访问方式,是若干访问方式的集合。矩阵中第 i 行第 j 列的元素 P_{ij}($P[S_i,O_j]$)表示主体 S_i 可以执行的对客体 O_j 的访问方式。比如 $P_{ij}=\{\text{read},\text{write}\}$表示 S_i 可以对 O_j 进行读和写的访问。

DAC 根据主体的身份及允许访问的权限进行决策。自主是指具有某种访问能力的主体能够自主地将访问权的某个子集授予其他主体。在 DAC 系统中,由于 DAC 可以将访问权限进行传递,对于被传递出去的访问权限,一般很难进行控制,这就造成资源管理分散,授权管理困难;用户间的关系不能在系统中体现出来;信息容易泄漏,无法抵御特洛伊木马的攻击;系统开销巨大,效率低下的缺点,不适合大型网络应用环境。

2. 强制访问控制

强制访问控制(MAC)根据中央权威所确定的强制性规则来进行访问控制。和 DAC 不同,强制访问控制并不具备访问主体自主性,主体必须在由中央权威制定的策略规则约束下对系统资源进行访问。强制访问控制是一种不允许主体干涉的访问控制类型,是基于安全标识和信息分级等信息敏感性的访问控制。在 MAC 中,系统安全管理员强制分配给每个主/客体一个安全属性,强制访问控制根据安全属性来决定主体是否能访问客体。安全属性具有强制性,不能随意更改。

MAC 最早出现在美国军方的安全体制中,并且被美国军方沿用至今。在 MAC 方案中,每个目标由安全标签分级,每个对象给予分级列表的权限,分级列表指定哪种类型的分

级目标对象是可以访问的。典型的军事系统采用的分级标签是不保密,限制,秘密,机密,绝密。典型安全策略就是 read-down(不向上读)和 write-up(不向下写)。

MAC 通过基于格的非循环单向信息流策略来防止信息的扩散,抵御特洛伊木马对系统保密性的攻击。系统中,每个主体都被授予一个安全证书,而每个客体被指定为一定的敏感级别。MAC 的两个关键规则是:不向上读和不向下写,即信息流只能从低安全级向高安全级流动。任何违反非循环信息流的行为都是被禁止的。

MAC 实现一般采用安全标签机制,由于安全标签的数量是非常有限的,因此在授权管理上体现为粒度很粗。但是由于 MAC 本身的严格性,授权管理方式上显得刻板,不灵活。如果主体和权限的数量庞大,授权管理的工作量将会非常大。

在 MAC 中,允许的访问控制完全是根据主体和客体的安全级别决定。其中主体(用户、进程)的安全级别是由系统安全管理员赋予用户,而客体的安全级别则由系统根据创建它们的用户的安全级别决定。因此,强制访问控制的管理策略是比较简单的,只有安全管理员能够改变主体和客体的安全级别。

MAC 应用领域也比较窄,使用不灵活,一般只用于军方等具有明显等级观念的行业或领域;虽然 MAC 增强了机密性,但完整性实施不够,它重点强调信息向高安全级的方向流动,对高安全级信息的完整性保护强调不够。

3. 基于角色的访问控制

随着网络技术的迅速发展,对访问控制提出了更高的要求,传统的访问控制技术(DAC,MAC)已经很难满足这些需求,于是提出了新型的基于角色的访问控制(RBAC)。RBAC 有效地克服了传统访问控制技术的不足,降低了授权管理的复杂度,提高了系统安全性。

一个角色定义为一系列权利,任何属于该角色的用户都享有这些权利。当访问系统时,每个用户必须明确他想扮演的角色,如果他被授权于那个角色,他便可以享有与该角色相应的权利。访问控制策略通过两个步骤实现:首先管理员定义角色和与之相应的权限,然后分派给每个用户一系列需要扮演的角色。角色可以按照等级设置,按等级活用访问控制权利的传递。

用户可能同时被授予多个角色,多个用户也可能拥有同一角色,即使安全管理员可能强行限制数目。

注意用户角色和用户组是两个不同的概念。组是用户的集合,而角色是权力的集合。进一步说,用户可以自由决定激活还是不激活角色,但是组成员只能是激活状态。

相对于 DAC 和 MAC,RBAC 的主要优点是它更适用于商业环境。事实上,在一个公司中,对系统的访问权限起重要作用的不是用户的身份而是他的职责。而且,在角色体制的访问控制中,基于角色的策略试图根据组织的结构构造权限。

2.2.3 访问控制的原则

授权过程和访问控制通过授权数据结构的使用,清楚地定义谁能够使用系统的什么资源以及系统的哪些资源不可用。系统资源通过最小特权和职责分离来进行保护,这最终导致责任的提高,从而提高了系统安全。

1. 最小特权

最小特权原则要求主体按照自己的需要获得授权。最小特权原则基于两条原则：最小权限原则和最小风险原则。基本的思想是使用系统资源的主体获得的权限不多于他们请求执行的任务的最小值和执行任务所需要时间的最小值。最小特权原则可以降低未经授权访问系统带来的风险。

在日常生活中，最小特权的例子也很多。例如，一些汽车制造厂制造汽车锁，用一把钥匙开车门和点火器，而用另一把钥匙开手套箱和衣物箱；停车场的服务员有安排停车的权力而没有从汽车行李箱里取东西的权力。

在信息系统安全设计中，需要最小特权的例子也很多。例如每个用户并不需要使用所有的网上服务，不需要去修改(甚至去读)系统中的每一个文件，也不需要都知道系统的根密码(root password)；每个系统管理员也没有必要都知道系统的根密码；每个系统也不需要去申请其他系统的文件。

2. 职责分离

职责分离是指遵循不相容职责相分离的原则，实现合理的组织分工。例如一个公司的授权、签发、核准、执行、记录工作，不应该由一个人担任。职责分离将不同的责任分派给不同的人员以期达到互相牵制，消除一个人执行两项不相容的工作的风险。例如收款员、出纳员、审计员应由不同的人担任。计算机环境下也要有职责分离，为避免安全上的漏洞，有些许可不能同时被同一用户获得。

2.2.4　多级安全模型

多级安全(Multilevel Security，MLS)的概念始于 20 世纪 60 年代，当时美国国防部决定开发一些保护计算机中存储的机密数据的措施。在这之前，一直是采用一些规章制度来限制那些未经批准的人员处理系统内的机密数据，因为那时还没有值得信任的计算机能够有效地保护计算机内的机密数据。在多级安全系统中，所有信息都有一个密级，每个用户也都相应地有一标签。要决定是否允许某用户读一个文件，那么就比较该用户的标签是否与该文件的秘密级相符。安全策略要求，为了合法地得到某一信息，用户的安全级必须大于或等于该信息的安全级，并且该信息属于用户的信息访问类别。

美国国防部(DoD)将秘密等级由高到低分为四个级别：最高机密(TOP SECRET)、机密(SECRET)、秘密(CONFIDENTIAL)、非秘密(UNCLASSIFIED)。不妨假设一个主体是 SECRET 级别，那么它可以访问 SECRET 级别(或更低级别)的客体，但是不可以访问 TOP SECRET 级别的客体。注意由于某些未知的原因，安全级别一般用大写表示。

假设 O 是客体，S 是主体。O 的安全级别用 L(O)表示，S 的安全级别用 L(S)表示。在 DoD 系统中，它们都使用上面四个级别，而且，对于一个 SECRET 级别的对象，或多或少要求进行常规的背景资料检查，而 TOP SECRET 的对象则要求扩展背景资料检查和测谎测试。

有许多实际问题和信息分类相关。举一个例子，分类区间并不总是清楚的，两个有经验的用户可能有很多观点不同。而且，采用的安全级别的粒度也是一个问题，完全有可能创建一个文档，它的每段是 UNCLASSIFIED，而整个文档是 TOP SECRET。当源程序需要分

类(DoD 有时会发生这种情况)时,这个问题变得更为严重。与粒度相反的是聚合,敌手可能通过仔细分析 UNCLASSIFIED 文件来搜集 TOP SECRET 信息。

当主体与客体是不同的安全级别而使用相同的系统资源时,多边安全就十分重要了。MLS 系统的目的是加强访问控制的形式,通过限制主体对客体的必需的访问级。军事和政府一直以来对 MLS 很感兴趣,特别是美国政府,已经进行了很多 MLS 的研究,因此,MLS 的优点和弱点相对比较清楚。

1. Bell-LaPadula 模型

这里首先介绍的安全模型是 Bell-LaPadula(BLP),这个模型是以它的发明者 Bell 和 LaPadula 来命名的。BLP 的目的是当涉及任何 MLS 需要满足的秘密性时,获取最小的请求。首先讨论下面两个概念。

简单安全条件:主体 S 可以读客体 O,当且仅当 L(O)小于等于 L(S)。

* 属性:主体 S 可以写客体 O,当且仅当 L(S)小于等于 L(O)。

简单安全条件仅仅表明 Alice 如果缺少合适的访问级,就不能读文档。这个条件显然是每个 MLS 系统都要求的。

* 属性的设计是为了防止 TOP SECRET 信息被写入 SECRET 文件。这会破坏 MLS 的安全性,因为用户具有 SECRET 权限能够读 TOP SECRET 信息。写可能发生,比如计算机病毒的结果。注意,计算机病毒能够破坏 MLS 系统,这种攻击至今对 MLS 仍是很大的威胁。

当 * 属性暗示“不下写”时,简单安全条件可以被总结为“不上读”。因此,BLP 可以简单的表示为“不上读,不下写”。很难想象还有比这更简单的安全模型。

BLP 很简单,它是少数能够证明系统的模型之一;但是,BLP 太简单了,以致没有实用价值。

2. Biba 模型

BLP 处理秘密性,Biba 处理完整性。事实上,Biba 模型本质上是 BLP 的完整版。

如果相信客体 O_1 的完整性但不相信客体 O_2 的完整性,那么如果客体 O 是由 O_1 和 O_2 组成的,则不能相信客体 O 的完整性。换言之,O 的完整级别是 O 所包含的所有客体的完整性的最小值。对于秘密性,采用高水位线原理,而对于完整性,可以采用低水位线原理。

为了更好地说明 Biba 模型,不妨令 I(O)表示客体 O 的完整性,I(S)表示主体 S 的完整性。Biba 模型有下面两个概念需要理解:

写规则——主体 S 可以写客体 O 当且仅当 I(O)小于等于 I(S);

读规则——主体 S 可以读客体 O 当且仅当 I(S)小于等于 I(O)。

Biba 模型表明完整性高的主体可以写完整性低的客体,完整性低的主体可以读完整性高的客体。从本质上说,我们关心的是 S 是否被低完整性数据“污染”,例如老师(高完整性主体)可以修改(写)小学生的作文(低完整性客体),小学生(低完整性主体)可以学习(读)老师的范文(高完整性客体)。

图 2-10 说明了 BLP 和 Biba 模型的不同。最根本的不同是 BLP 的目的是秘密性,采用高水位线原理,而 Biba 的目的是完整性,采用低水位线原理。

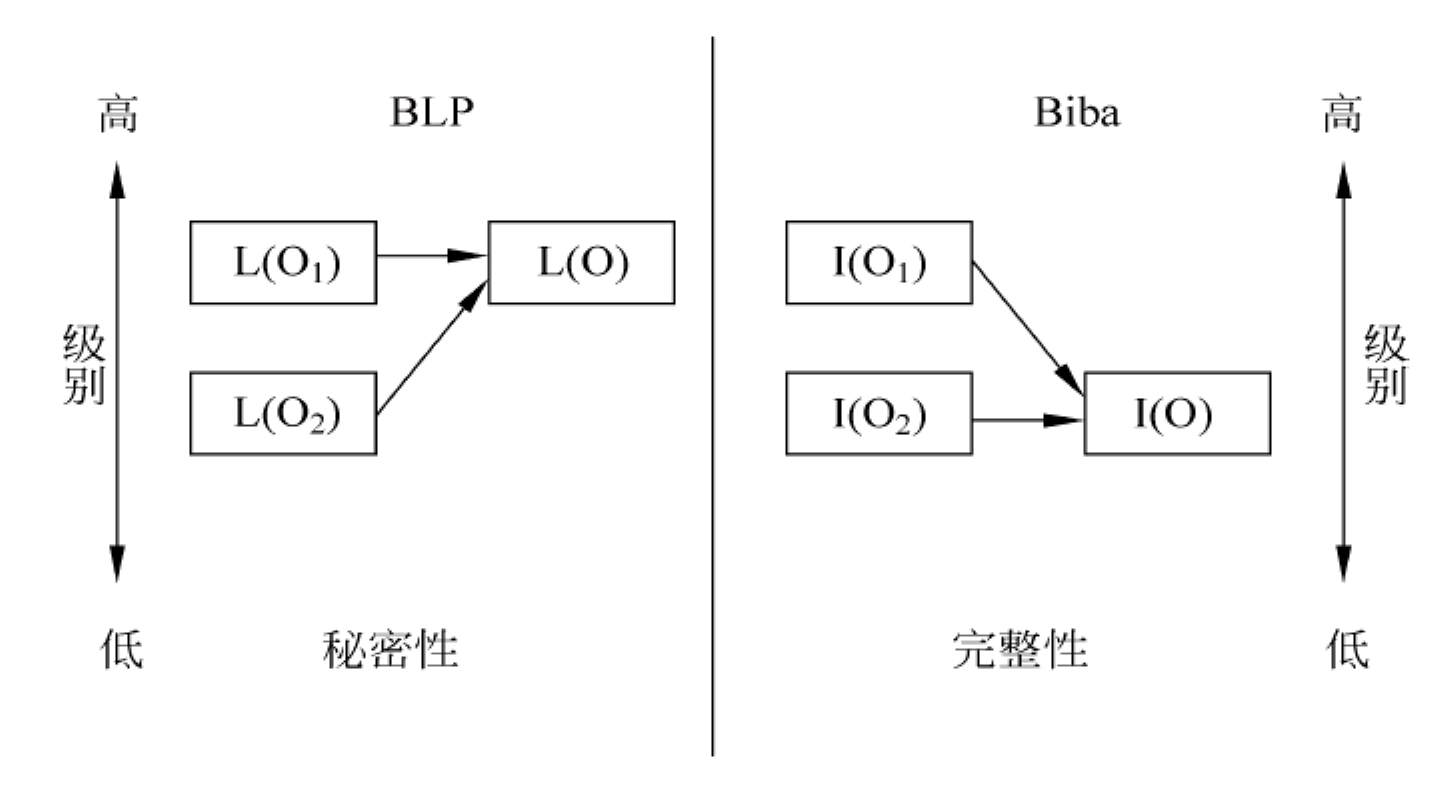


图 2-10　BLP VS Biba

2.2.5　多边安全

多级安全系统加强了访问控制(或者信息泄露)“起伏变化”,多级系统中安全级别是按照等级设置的。一般而言,安全标签的简单等级对于处理实际环境并不十分灵活。多边安全使用区间(compartment)进一步限制了信息通过安全等级。

使用概念:安全等级{区间}来表示一个安全等级以及与之相关的安全区间。举一个例子,如果有区间 CAT 和 DOG 属于 TOP SECRET,则可以表示为 TOP SECRET{CAT},TOP SECRET{DOG}和 TOP SECRET{CAT,DOG}。当每一个区间都是 TOP SECRET 时,如果允许 TOP SECRET 级别的主体 S 访问区间,那么它仅能够访问一个区间。因此,区间限制了信息通过安全等级。

区间用来加强“需要知道(need to know)”原则,即主体仅允许访问他们必须知道的信息。如果一个主体对 TOP SECRET 等级没有合法的“需要知道”,那么区间可以用来限制主体可以访问的 TOP SECRET 信息。

为什么 MLS 模型中用区间代替简单创建一个新的分类等级呢?原因是这样的,举一个例子,TOP SECRET{CAT}与 TOP SECRET{DOG}是不可比的,即 TOP SECRET{CAT}小于等于 TOP SECRET{DOG}与 TOP SECRET{CAT}小于等于 TOP SECRET{DOG}都不成立。使用严格的 MLS 等级,这两个式子可能有一个成立。

图 2-11 中是一个多边安全模型,箭头表示“大于等于”的关系。在这个例子中,主体有 TOP SECRET{CAT}级别的权限,但不能访问 TOP SECRET{DOG}区间的信息。另外,如果主体有 TOP SECRET{CAT}级别的权限,那么它能访问 TOP SECRET{CAT}区间的信息,但不能访问 TOP SECRET{CAT,DOG}区间的信息,即使该主体有 TOP SECRET 级别的权限。可知,区间的目的是加强“需要知道”原则。

没有多边安全(区间),多级安全也能使用,反之亦然,但是它们通常一起使用。比如英国医协会(British Medical Association,BMA)的法律系统使用的是多级安全系统——因为法律制定者对 MLS 很熟悉。一些疾病(比如 AIDS)被认为是 TOP SECRET,而不太敏感的信息(比如处方)被认为是 SECRET。但是如果主体已经开了治疗 AIDS 的药物,任何拥有 SECRET 权限的人很容易就能够推测到 TOP SECRET 信息。因此,所有的信息有必要以更高的级别分类,这样所有的用户都要求高级别的权限,这就违背了系统的目的。最终,

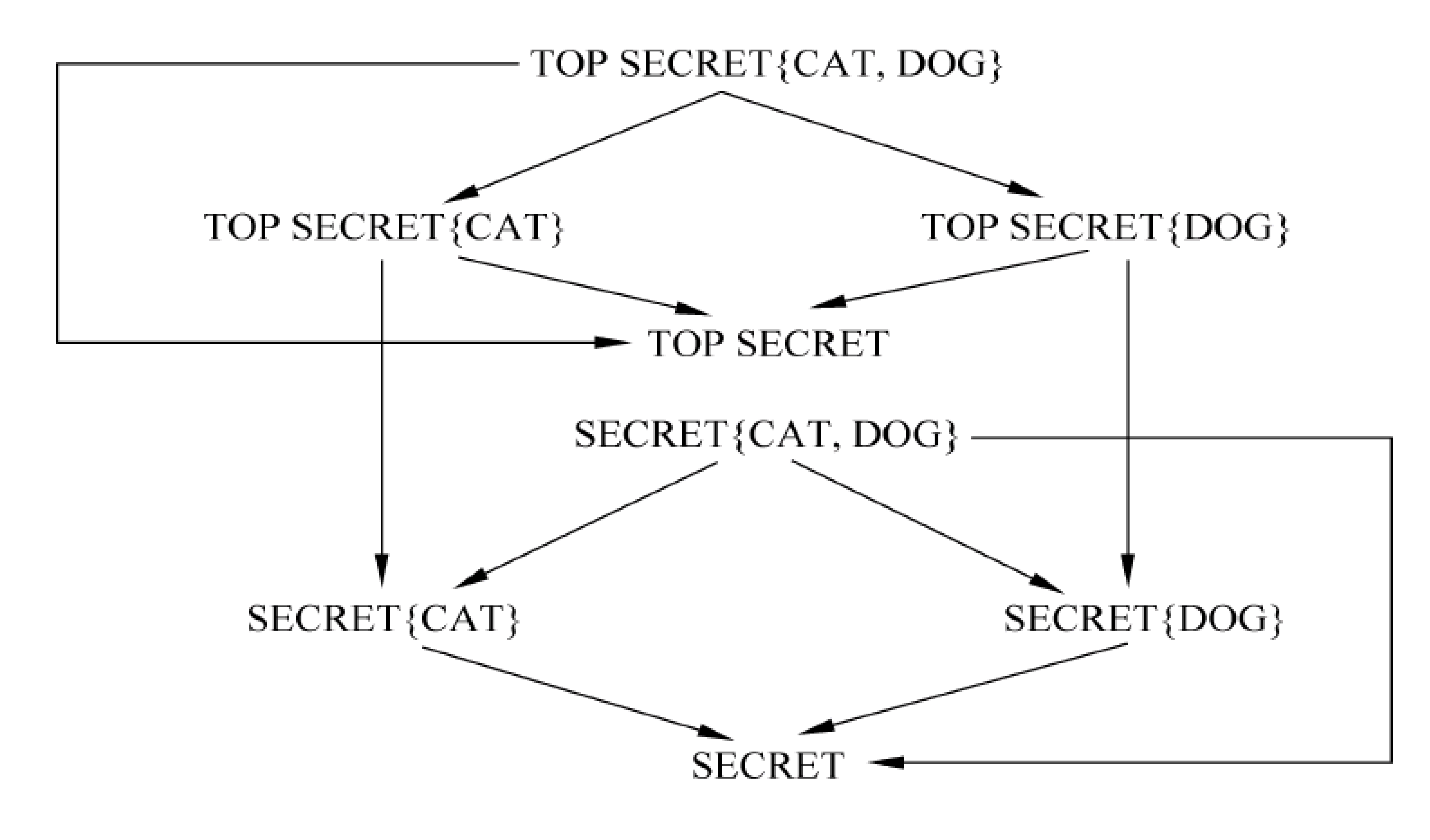


图 2-11 多边安全例子

BMA 系统使用了多边安全系统，从而有效地解决了问题。于是，AIDS 处方信息从一般处方信息中间隔出来，从而加强了“需要知道”原则。

2.2.6 隐蔽信道

隐蔽信道可以简单描述为不是系统设计者设计的通信路径。隐蔽信道在许多情况下会发生，特别是网络通信中。隐蔽信道在现实中几乎是不能消除的，这里的目的是要限制这些信道的容量。

设计 MLS 系统的目的是限制合法信道的通信，但是隐蔽信道提供了信息泄露的另一种方法。举一个例子，共享资源的不同安全级别的主体可以用来传递信息，这就违背了“no read up，no write down”的 BLP 限制。

假设 Alice 拥有 TOP SECRET 级别的权限，而 Bob 只有 CONFIDENTIAL 级别的权限。如果所有用户共享文件空间，Alice 与 Bob 可以协商，如果她想发送 1 给 Bob，她需要创建一个文件 FileXYzW，如果她想发送 0，则不需要创建这样的文件。Bob 检查 FileXYzW 是否存在，如果文件存在，他知道 Alice 已经发送给他 1；如果不存在，则传给他的是 0。通过这种方法，单比特信息可以通过隐蔽信道传输，即通过一种并不是系统设计者用来通信的方法。注意 Bob 不能知道文件 FileXYzW 的内容，因为他没有要求的权限，但是假设他能够询问文件系统是否存在这样的文件。

从 Alice 到 Bob 的单比特泄露可能不会引起注意，但是通过与 Bob 同步 Alice 能够泄露任何数量的信息。比如，Alice 与 Bob 可以商议让 Bob 每分钟检查一次 FileXYzW 文件，如果文件不存在，那么 Alice 发送的是 0，如果存在则 Alice 发送的是 1。通过这种方法 Alice 可以慢慢将 TOP SECRET 信息泄露给 Bob。图 2-12 说明了这个问题。

隐蔽信道有多种方法建立，比如打印队列，可以用来标志信息，方法与上面的文件例子相同。网络流量有许多潜在的隐蔽通道，事实上，在一些黑客工具中也能够发现隐蔽信道。

存在隐蔽信道需要三个条件：第一，发送方和接收方必须访问共享资源；第二，发送方必须能够区分接收者注意到的共享资源的属性；第三，发送方与接收方在通信中必须同步。由此可知，隐蔽信道十分普遍，大概完全清除隐蔽信道的方法只有清除所有的共享资源和所

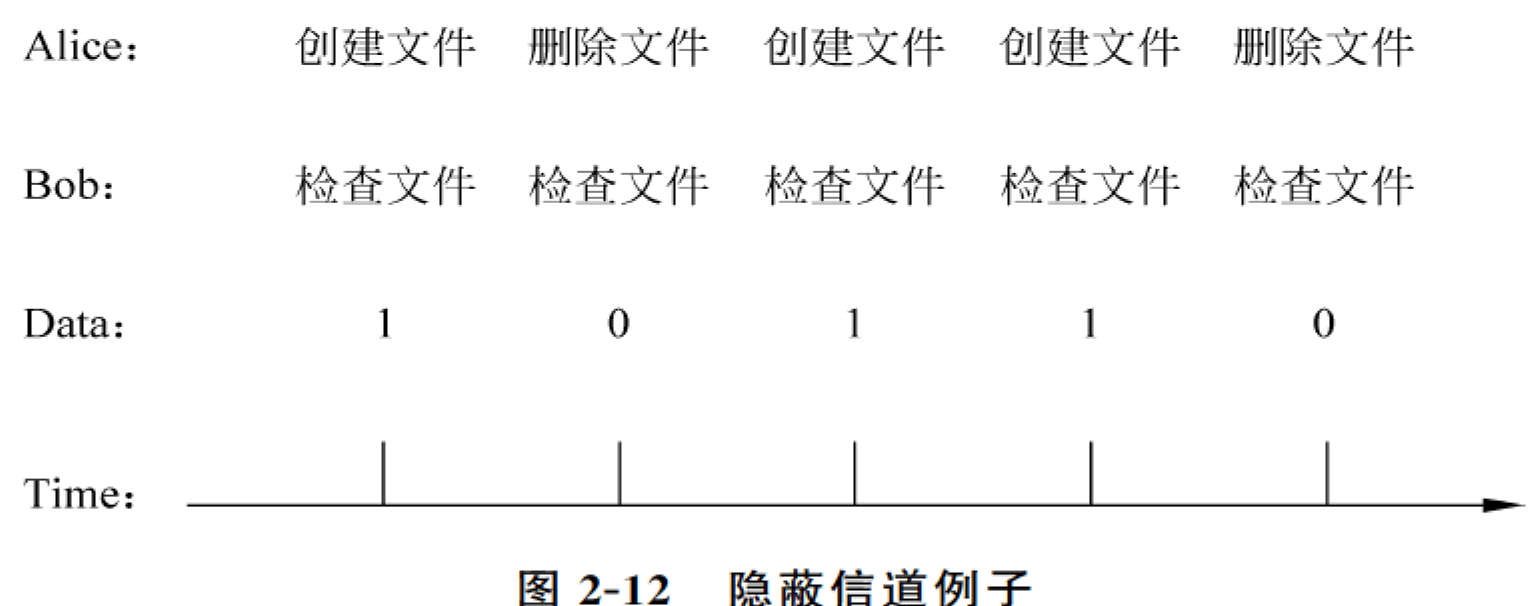


图 2-12　隐蔽信道例子

有的通信,这几乎是不可能的。

每秒 1 比特的限制足够隐蔽信道保护了吗?假设 TOP SECRET 文件是 100 MB,这个文件的明文版本存储在 TOP SECRET 文件系统中,加密版本的文件存储在 UNCLASSIFIED 位置。如果能够将这个系统的隐蔽信道容量设置为每秒 1 比特,那么通过隐蔽信道需要超过 25 年的时间来泄露 100MB 的文件。换言之,降低隐蔽信道的容量是很有用的,但是是不可能产生效果的。

举一个真实世界中隐蔽信道的例子,假设传输控制协议(Transmission Control Protocol,TCP)的头部包含用不到的"预订"域,这个域完全可以用来隐蔽地传递信息。也可以将信息隐藏在 TCP 序列号中或 ACK 域中,这样也可以在发送方与接收方之间产生一个隐蔽信道。图 2-13 说明了这个方法,通过 Covert-TCP 工具在序列号中传递信息。发送方在序列号 X 中隐藏信息,产生的数据包(它的源地址伪造成接收方的地址)被发送给任何服务器。当服务器收到数据包时,它不经意地完成了隐蔽信道——传递包含在 X 中的信息给了接收方。在网络攻击中这种隐蔽通道常常被用到。

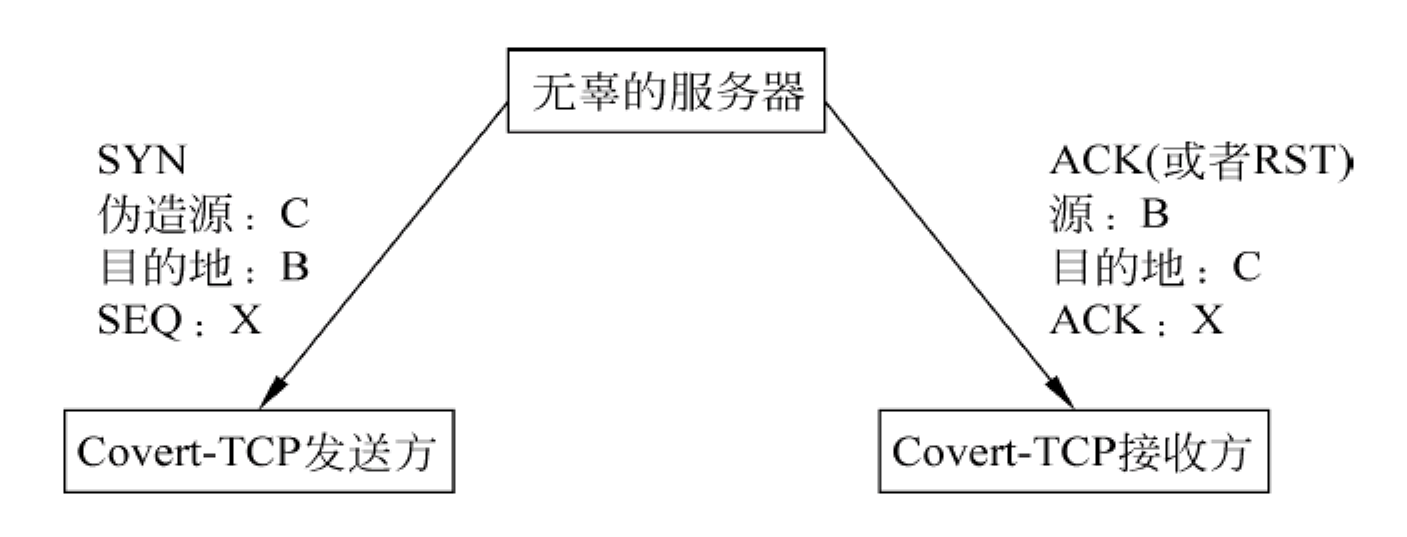


图 2-13　使用 TCP 序列号的隐蔽通道

2.3 机　密　性

机密性的目的是保护信息以免泄露或暴露给那些未被授权而拥有这个信息的人(例如,个人或组织)。信息可用许多不同的形式进行存储和传递,从而可通过不同类型的渠道泄露出去,这些渠道包括:①解释一个表达某一信息的数据项;②观察一个数据项是否存在(不管它的内容);③观察数据项的大小;④观察包含在数据项特征中的动态变化,如内容、存在性、大小。机密性服务潜在地需要保护所有这些泄露信息的渠道。

2.3.1 提供机密性的方法

有两种基本方法可提供机密性：①访问控制方法，防止入侵者观察敏感信息的表示；②信息隐藏方法。可允许一个人观测信息的表示，但他从表示中无法推出所表示的信息内容或提炼出有用的信息。

访问控制方法包括以下一些策略：

(1) 访问控制机制，它过滤每个寻找读访问信息资源的请求；

(2) 流控制措施，它防止信息流从一个(敏感的)环境流动到另一个(保护级别低的)环境；

(3) 物理安全措施，它防止物理上侵入一个有敏感信息的环境；

(4) 辐射安全保护，它防止电磁场振动的发射以免敏感信息被析出；

(5) 信息分离技术，例如，通过多个独立的通道传输不同的数据项的分量；

(6) 保护传输技术，如扩频技术或跳频技术。

在上面的这些对策中，影响到网络协议的对策是访问控制机制和流控制机制措施。访问控制机制在 2.2 节中已作了讨论。流控制关心的是传播信息的能力，它确定信息可以流动的合法信道。

隐藏信息的方法在对信息表示没有可靠形式的访问时需要，例如，当信息通过无线电或别的不可信通信设备时。加密是数据通信中隐藏信息的一种主要机制，不过由于实际需求，一些新技术如数字水印技术、量子密码技术、匿名技术等正在发展和成熟。

当信息在不同的环境之间运动时，可能需要不同的方法来实现机密性，图 2-14 就是一个例子。来自一个使用访问控制方法保护的环境的信息项，再使用隐藏信息的方法已经没有必要。当它运动到一个不同的没有可靠的访问控制保护的环境(例如一个公开网络)时，要在这个环境中维持机密性，就必须使用加密(隐藏信息的方法)。紧接着数据项又运动到另一个访问控制方法保护的环境，这时就不需要隐藏信息的方法。

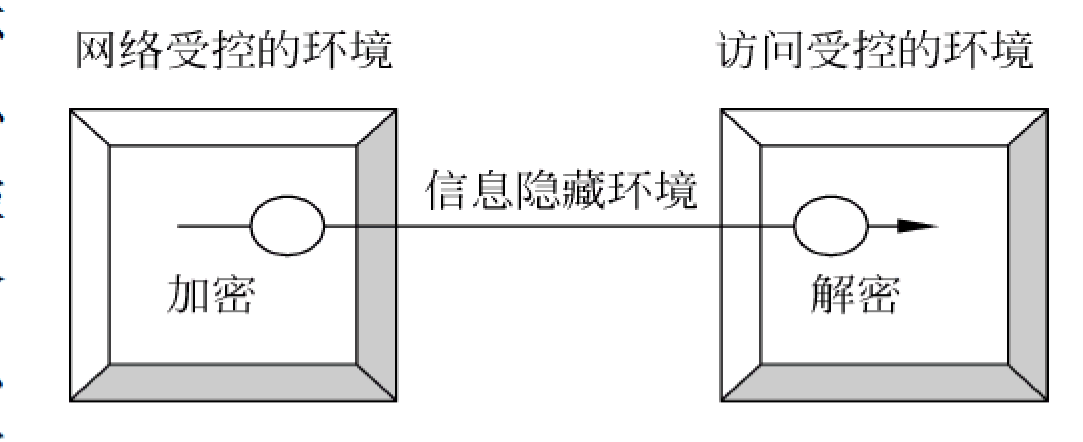


图 2-14 不同环境中的机密性保护

2.3.2 机密性机制

1. 加密

加密和解密提供了从明文流到密文流的一种交换方法，其基础在于加密和解密所用到的密钥，这是一种在信息隐藏中主要的提供机密性的方法。

加密机制的应用，需要考虑以下几个因素。

密码体制的类型：需要从对称密码体制和公钥密码体制中选择出一类。一般来说，公钥密码体制的密钥管理比前一种简单，但是在相同强度的条件下，公钥密码体制需要更多的资源。在带宽紧张的情况下，对称密码体制的效率要更高。

算法的选择：在每一类密码体制中，都有很多的密码算法。要从密码强度、性能、实现的代价、标准以及可出口性等方面来考虑。

工作模式：像 DES 这种固定分组大小的算法，对任意长度的数据需要一种工作模式。

填充需要：数据的大小经常不适合算法的需要，所以需要填充一些附加的数据。

初始化需要：在提供的机密性中，这里有一个一般的要求，加密一个数据项不总是产生同样的结果。否则，密文流中的重复模式往往能够提供更多的信息，即使不知道密钥。由于这个原因，链接和反馈密码体制使用初始向量 IV 来启动加密过程。每次的 IV 都不同，可以保密，也可以不保密。

同步需要：来自于同步密码体制。

密钥管理过程：密钥管理涉及各个不同阶段，包括密钥产生、密钥分配、密钥销毁、密钥取消等。

一些因素，如密码体制类型和密钥管理过程对应用和其环境是基础的，然而在不同的应用实例中这些细节可能很不一样。安全协议在实际应用中通常提供传送密码算法的标识符和参数。算法的参数包括：工作模式和（或）填充约定的标志；具体密码体制的参数，比如 RSA 中模的长度；初始化向量值；工作模式和（或）填充约定通过算法标识符来暗示，或者它们需要独立地确定。

2. 数据填充

为了防止通过监视传送数据的长度来获取信息，一般采用数据填充技术。比如，可以通过观测用户 A 和用户 B 今天传送的数据比通常更长，就有可能推出某些信息，即使不知道解密它们的密钥。数据填充和加密机制往往联系在一起，来保护数据不被泄露。通常，在加密前把明文的长度填充为规定的长度。

3. 业务流填充

业务流的机密性是为了防止通过观察网络业务流而使敏感信息泄露。例如，某人可通过观察发现白宫和五角大楼之间某天的消息传输比平时繁忙了，从而得到一些信息，而完全不需要知道这些信息的内容。而且，如果窃听者能知道这些增加的消息是从白宫的某个机构到五角大楼的某个机构之间传输的，那么他就可以得到更多的信息。

业务流填充是提供业务流机密性的一个基本机制。它包含生成伪造的通信实例，伪造的数据单元和（或）伪造的数据单元中的数据。这个机制与加密无关，因为有必要隐藏真实数据和伪造数据之间的差别。然而，注意到业务流填充和资源共享（如在一个分组交换网络或一个局域网络中）存在着固有的矛盾。在一个通道中的业务流填充会导致通道总是满容量运行，从而大大地减少了资源共享的可能。

在一个资源共享的多节点的网络环境中，业务流机密性变得很难保证。通常，一个通信如建立连接或无连接数据单元的传输，需要传送一个能被中继系统解释的地址。依赖于所有网络组件的信任关系，这个地址信息的不同部分能被加密成不同的值。因此，地址隐藏成为多点和多节点协议中的重要研究问题。

4. 其他机制

除了上面介绍的机制外，还有一些其他提供机密性的方法，如路由控制、数据分割、扩频技术和跳频技术。

网络协议中的路由控制提供信息流控制。这种访问控制形式结合可信系统组件提供安全标记机制。数据分割能够用很多不同的方法实现。比如，要秘密地传送一个文件，可以把

文件比特序列分成不同的文件，其中，一个包含奇数位的比特的数据流，另一个包含偶数位比特的数据流。这两个文件能够独立地传送，比如，不同的通信系统，不同的路由和不同的时间。这种方法能有效地对抗多种类型的窃听威胁。

秘密共享是数据分割的一种变形。它能应用于密钥管理，比如当一个主密钥需要手工分配给一个系统时，将这个密钥分成两部分，由两个不同的人持有。使用系统将需要两个人合作，这就减小了单个人持有密钥滥用系统的机会。

扩频技术和跳频技术在商业和非机密性的环境中很少使用。

2.4 完 整 性

完整性的目的是保护数据以免被未授权的用户改变、删除或替代，它是对改变数据的价值和存在所采取的一类防护措施。

2.4.1 提供完整性的方法

数据完整性服务使数据不受无意或有意的损坏。有以下两种基本方法来保证数据的完整性。

（1）访问控制方法：用于避免入侵者有机会损坏数据；

（2）损坏-检测方法：承认数据会有损坏（有意或无意），但能确保这些损失会被检测出来，并且能够被纠正或向接收者提出警告。

利用访问控制方法来保证数据的完整性类似于机密性的情况。若数据在可信环境中应用，数据被有意地破坏的风险能被消除。数据的完整性相对而言比较容易实现。然而，要注意访问控制不能保护无意的数据损坏（例如在一个安全但有噪音的信道中）。

一个完整性服务需要保护数据在能应用的粒度上不能被创建或修改。注意到，在连接完整性的情况下，这个必要的保护用来防止在连接中传输的个人数据单元的重放、重排或丢失。

完整性服务没有必要保证在涉及的粒度上的数据单元不被重放、重排或丢失。例如，连接完整性服务没有必要保护整个连接不被重放，而无连接完整性服务没有必要保证特定的无连接信息单元不被重放、重排或丢失。更进一步的保护有时需要，且能用一种序列完整性机制来实现。

对数据损坏的检测有两种相应的方法。损坏能被报告给可能远离该过程或进行其他活动的接收者。相应地，系统又试图通过引起传输数据被破坏的方法自动地恢复破坏数据。后者需要一个附加的恢复机制。

2.4.2 完整性机制

1. 测试字

银行系统应用一种叫测试字（testword）的检测-损坏数据完整性机制已经很多年了。设计的这些系统应用于确保在不受保护的电报/电话网络上进行的金融交易。它们利用双方协定来使交易字段受到保护（例如账号名称、日期和总量），通过一个静态的内部规定的密

钥和一张包含着一串随机数的卡。发送者通过一个双方认同的算法将信息转化成一个字符串(测试字)附加在交易中,接收者用接收到的交易数据重复同样的步骤,从而能够验证交易的完整性。测试字是封装的一个早期技术实现。

2. 封装和签名

最普遍的封装和签名技术包含着用加密方法产生一个作为明文的附件传送的值。当需要特定的数据完整性的要求时,这样一个附件通常被称为完整性校验值(ICV)。

封装和签名机制的应用需要在使用者中解决几个问题,例如在算法的选择上,填充要求上和密钥管理过程上都需要建立共识。为机密性目的加密机制的使用的大多数讨论适用于数据完整性目的的封装和签名机制的使用。

3. 加密

一些加密机制既可用于保证数据的完整性,也可用于保证机密性。假定被保护的数据项拥有一些冗余,加密传输冗余能保证数据完整性。因为如果一个入侵者不知道加密的密钥而修改了密文的一部分,会导致在解密的过程中产生不正确的信息。

冗余可用多种途径得到。在一些情况下,数据项拥有充分自然的冗余。相应的,冗余能优先于加密通过扩展数据项得到,一个校验值称为修改检测码(MDC)。一个 MDC 的例子是一个由 Hash 函数产生的数据摘要,或一个循环冗余校验。

不是所有的加密机制都能提供完整性。例如,在加密模式中,一个可逆的公钥算法不能够保证数据的完整性。假定一个入侵者知道公钥,那么他能加密假的数据项。

4. 序列完整性

序列完整性为检测数据项的重放、重排或丢失提供了一种方法,它形成了一个序列的一部分。假定每一个数据项内部的完整性由上述的一种机制保护。有两种方法可以提供序列的完整性:①在通过封装、签名或加密等保护之前,给数据项附加一个完整性序列号(ISN);②在封装、签名或加密等过程中利用在数据项的序列上扩展的链产生一个加密链。这两种方法的典型过程如图 2-15 所示。

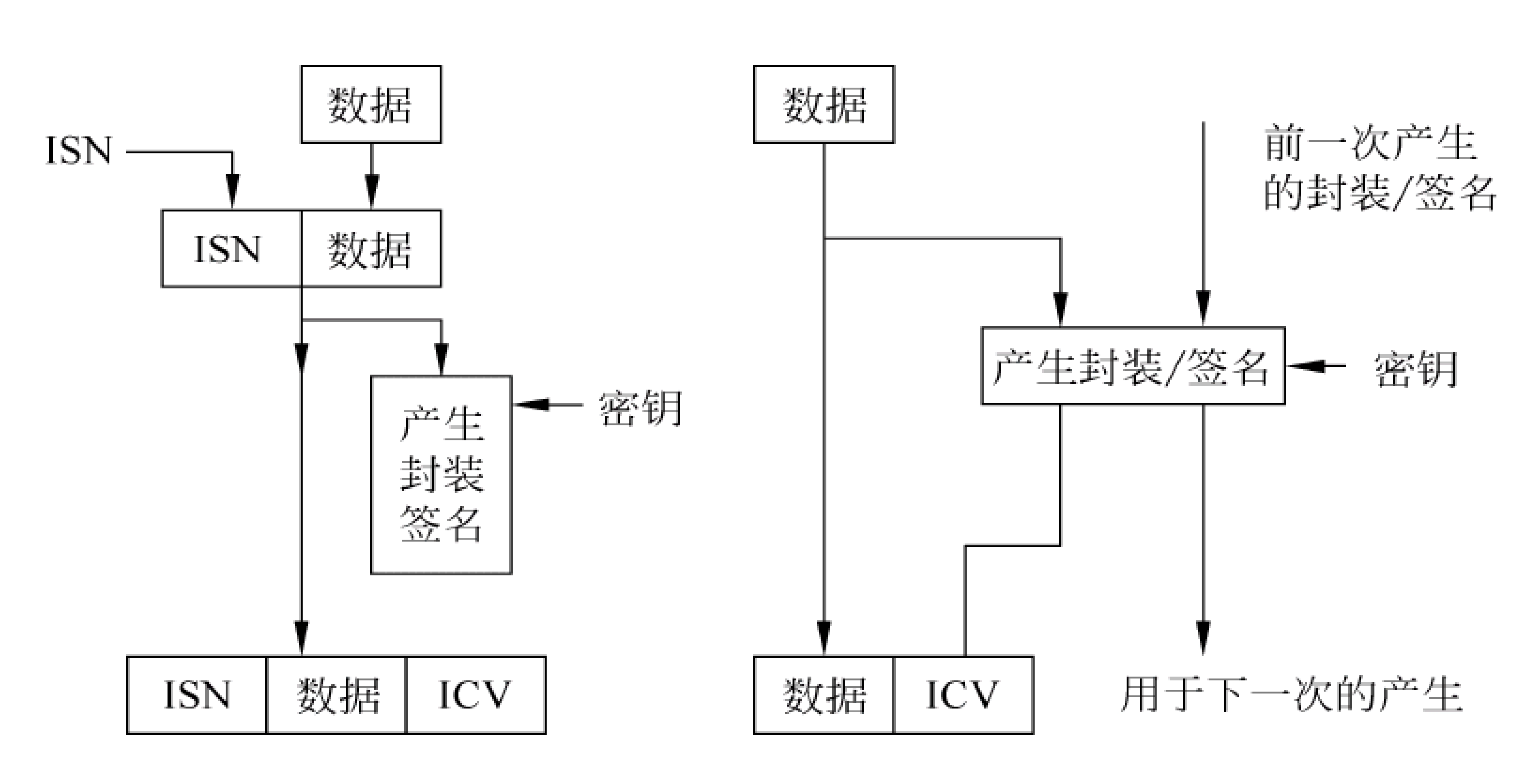


图 2-15　实现序列完整性的两种方法

5. 复制

一个数据完整性的度量能由多存储区域的存储信息或通过在不同的路径中传输数据的

多个备份来提供。可假想攻击者不能同时危及所有的备份,那么原始数据就可以从未被危及的备份中恢复。

6. 完整性恢复

支持完整性恢复所需的机制是一个简单的标准通信错误恢复机制。通常的方法是在检测到破坏发生和重新发送那个位置之后的所有数据之前重新同步到检测位置。任何密码处理过程,不管是为了机密性还是完整性,均需要同时重新同步到同一个检测位置。完全恢复的任何失败都可用检测原始失败的同样的完整性机制来检测。

2.5 不可否认性

非否认服务为一个通信用户提供保护以免另一个用户后来否认所发生的一些通信交换。当这些服务不能防止一个用户否认另一个用户所声称发生的某事时,它们确保可获得无可辩驳的证据以支持任何类似纠纷的迅速解决。通常,这样的证据必须能说服一个第三方仲裁者。

在数据网络环境中,否认情况可分成两种不同的情况:一是起源的否认,即存在着关于是否某方产生了关于某个特定数据项的纠纷和(或)关于产生时间的纠纷;二是传递的否认,即存在着关于是否某个特定数据项传递给了某方的纠纷和(或)关于传递发生时间的纠纷。

2.5.1 非否认过程中的各个阶段

与非否认相联系的事件序列可分为五个不同的行动阶段:服务请求,证据产生,证据传递/储存,证据验证和纠纷解决。

1. 服务请求

首要的是一个计划好的非否认服务的应用必须在关键行为(critical action,作为非否认服务目的的事件或行为,如一个特定数据项的产生或一个特定数据项的传递)发生之前被承认。假设 A 和 B 将要参与某个通信事件,同时 B 想获得 A 事后不能真实地拒绝承认所发生过的事实的保证。这一服务的提供需要 A 的合作,A 需要参加所需要的证据的产生。这可能需要 B 或者某一有关的第三方制定一个清楚的预先给 A 的请求:非否认将要应用于即将来临的这一特定事件中。当有一个标准的安排或通常策略存在于各方之间使得非否认永远应用于某一些类型的事件的时候,这种清楚的请求并不总是必需的。

作为一个纸上的交易模拟,假设关键行为是 A 立即支付一定数目的金钱给 B 的承诺,而且将被应用的这一机制就是传递那张支票。非否认机制就相当于在支票上的签名。通常,如果存在一个已建立好的协定使得任一开支票的那一方会被期望同时签上名,B 清楚地要求 A 在支票上签名不是必需的。然而,在某些情况下,B 或 B 的律师会清楚地要从 A 那里得到一个特定类型的非否认,例如,一张已被鉴定的支票。

与这一阶段联系的角色是一个服务请求者和一个证据产生者。服务请求者常常是直接参与关键行为的通信方中的一个,但可以另选有密切关联的一方,如一个安全区域中的权威

机构。在那个支票传递模拟中，服务请求者可假定为 B 或 B 的律师。

2. 证据产生

证据产生与关键行为相关联，而且由一方或者操作一致的某几方组成的群体来完成，它或它们扮演着证据产生者的角色。被认为是关键行为的潜在否定者的那一方需要参与证据的产生。可信第三方也应该参与。

回到上面支票的支付模拟中，证据产生包括生成那个必需的支票签名和一个银行证明（如果需要的话）。

3. 证据传递/存储

关键行为发生后，证据已经产生，证据要被传递到最终需要使用的一方或几方，和（或）被可信第三方存储可能作为将来的参考。

在支票支付的情况下，证据以签名的形式传递给最终用户，并且如果需要，支票文档中的银行证明也要传递，银行也要保存证明的一个备份。

4. 证据验证

证据验证就是核对被传递的或被可信地保存着的证据。这一阶段是事件序列的一个正常部分，它不是一个纠纷的结束。它的目的是让各方信任被提供的证据将会足以应付纠纷引发的事件。

在支票支付的情况下，这一阶段包括接收者的各种检查，如文档是否被适当的一方签名，日期是否正确，是否被银行完全地证明了（如果必要的话）。如果接收者熟悉支付方的签名，它会扩展到扮演一张有个人签名的支票。

这一阶段引进一个新的角色——证据验证者。这一角色被假定为单独的一方或者操作一致的多方。一般包括提出非否认服务请求的一方和（或）可信第三方。

5. 纠纷解决

最后的阶段就是解决纠纷的阶段。它将发生在与前面阶段完全不同的事件框架里。它确实可以发生在那些阶段之后不久（如同在同一个通信会话中），但更可能发生在之后很久（如几年以后）。

如果这一阶段发生，必须找回证据，重新核实它，然后解决纠纷。这需要引进另一个角色——仲裁者，这个角色被假定为一个第三方，如法庭。

2.5.2　起源的非否认

起源的非否认关系到关于是否某一特定方“产生（originate）”了一个特定数据项的潜在的争执和（或）关于产生时间的争执。由于非否认这一形式，证据产生者包括了发起者和在某些情况下一个或几个可信第三方。服务请求者包括了数据的潜在的接收者和（或）代表了接收者利益的那几方。证据验证者包括数据接收者和（或）可信第三方。

证据要求能够把各种信息片段无可辩驳地联系或连接起来，至少包括：①发起者的身份；②数据项的精确值。在某些情况下，也包括其他信息，例如：①“产生发生的日期和时间”；②意定的接收者的身份；③与证据产生有关的所有可信第三方的身份。

这里集中于非否认在一个数据通信内容中的起源的非否认上，该服务和大多数实现它的机制同样适用于数据项的产生和紧接着的数据库存储。在这种情况下，重新获得数据项

的一方代替了原来在通信内容中被认可的接收者。常用的可以提供起源的非否认的基本方法有以下几种。

1. 发起者的数字签名

一个提供起源的非否认的机制是发起者数字地签署数据项并将签名传递给接收者。数字签名组成了主要的证据。证据验证阶段接收者验证数字签名,此后数字签名被接收者保存。若以后发生发起者否认数据项起源的事件,接收者可以向对方出示保存的数字签名作为无可辩驳的证据。

假设一个基于公钥密码的数字签名方案被使用,可以考虑扮演可信第三方的那个角色。检查和保留公钥证书是这一过程的最基本的部分。如果在纠纷解决中接收者由于相信自己有正确的公钥而生成包含一个数字签名的证据,这还不足以说服第三方仲裁者。公钥的合法性的确凿证据必须来自可信第三方。

验证过程必须使用公钥证书,公钥证书从经过认可的认证机构那里得到,该机构在纠纷引起时可以再证实证书和证书内容的合法性。因此,发布公钥证书以满足非否认意图的认证机构的责任会稍高于那些发布公钥证书以满足基本认证、机密性或完整性意图的机构的责任。

公钥撤销时需要更多的注意。如果公钥证书被撤销,这时撤销的精确时间必须让所有证据验证者知道。被证明过的密钥在撤销前产生的证据是合法证据,而撤销后产生的证据则是不合法的。在它们的验证过程中,证据验证者必须极其严格地检查是否存在可能的密钥或证书的撤销。此外,非否认可以要求实时撤销,这比实现通常的周期的分布式撤销列表更加困难。

证据的验证和存储的密钥同时被撤销的可能性的大小显然是个问题。这种事件的结果如何取决于非否认的策略。这使得发起者保存适当的被证明过的撤销列表以支持仲裁者那里关于发起者的情况是有必要的。

2. 可信第三方的数字签名

一个可选的机制采用可信第三方的数字签名代替发起者自己的数字签名。可信第三方可以看做是发起者的担保。通过这一机制,发起者传递数据项给可信第三方,可信第三方对数据项、发起者身份以及所有其他必需的信息(如时间戳)产生一个数字签名(称为对数据的可信第三方的数字签名)。这个签名成为证据,被传送给接收者,接收者对其验证并存储。

从发起者到可信第三方的通信要求对完整性和数据起源的认证保护,使得可信第三方可以确信它签了一个具有合法价值的信息。从可信第三方回到发起者的通信则不一定要完整性保护,因为发起者有机会在传送给接收者时自己核实数字签名。

这个机制与发起者的数字签名机制相比有两个优点:一个优点是验证常常涉及较少的密钥对,因此减少了对证书获得、核实和储存的需求,一组数目相对少的签名者们能支持发起者和接收者的大量通信,这些通信为验证者提供了更大的潜力去保留和重用被证明过的公钥;另一个优点是取得了可信第三方签名可以认为维持了一个可信的时间源,时间戳可包含在签名里头,这就为接收者提供了一个发起者的确定时间。

作为以上方案的一个变形,发起者用 Hash 函数生成数据项的摘要,将摘要发送给可信第三方,然后可信第三方对摘要、发起者身份和所有其他必需的信息(如时间戳)产生一个数

字签名(称为对消息摘要的可信第三方的数字签名)。这个数字签名成为证据返回给发起者,然后转发给接收者。

用数据项的摘要代替数据本身来生成签名有几个优点:采用摘要的方法减少了要被送到可信第三方的数据量;它能够保留在发起者和可信第三方之间通信的原数据的机密性,而且保留在可信第三方的系统内。

如果存在发起者使用多种不同 Hash 函数的可能性(而且这是个合理的假定,因为考虑到合理的 Hash 函数定义的易变性),这时摘要总是相当于一个特定 Hash 函数的明确的标识符就显得特别重要了。这个标识符必须被包括在可信第三方签署的值里,否则会存在一个有争议的情况,发起者产生两个不同的数据项,用两个不同的 Hash 函数生成同样的摘要。如果这时候没有将摘要和 Hash 函数的标识符安全地联系在一起,则签名将不会提供关于在该特殊情况下生成的数据项的无可辩驳的证据。

3. 可信第三方令牌

可信第三方令牌是可选的可信第三方签名方案,签名方案用对称密码技术代替了公钥密码技术。令牌类似于一个签名,除非它没有被签署,否则用一个对称算法把它封装起来而且密钥只让可信第三方知道。包含在封装计算中的信息包括数据项或其摘要、发起者身份和其他需要的信息(如时间戳)。令牌由可信第三方为发起者生成,传送到接收者那里。与前面方案的不同之处在于接收者不能直接验证令牌,但可送回到可信第三方处验证。

发起者与可信第三方之间,以及接收者与可信第三方之间的通信交换都需要完整性和数据起源的认证保护。

4. 内联可信第三方

另一种机制包含了可信第三方如何插入到发起者和接收者之间的通信线路中。可信第三方可用多种方式操作,最简单的方式如图 2-16 所示。可信第三方只是简单地捕捉和储存足够证据以支持解决未来有关通行的数据的起源的争执。这种证据通常包括数据的副本或摘要、发起者身份和时间戳。

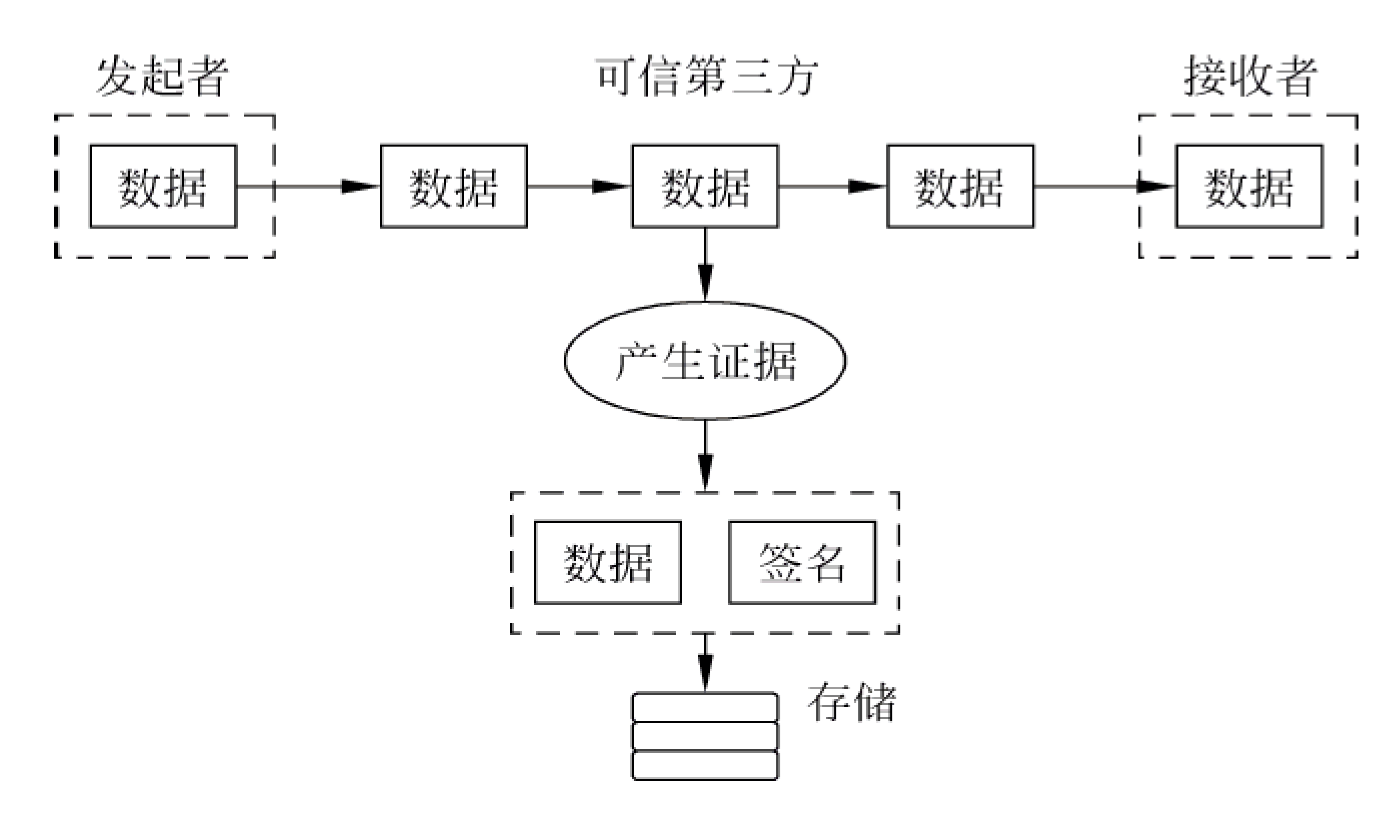


图 2-16 内联可信第三方——证据存储

这种机制有与应用协议字段的格式不冲突的优点,但它依赖于其他安全服务的使用。特别地,完整性和数据起源认证服务必须应用在发起者与可信第三方以及可信第三方与接

收者的通信中。

另一种内联可信第三方如图 2-17 所示。在这里,可信第三方生成一个摘要作为证据转发给接收者用于验证和存储。它类似于较早时描述的可信第三方签名机制,除了在发起者和可信第三方之间必须使用完整性和数据起源认证服务。

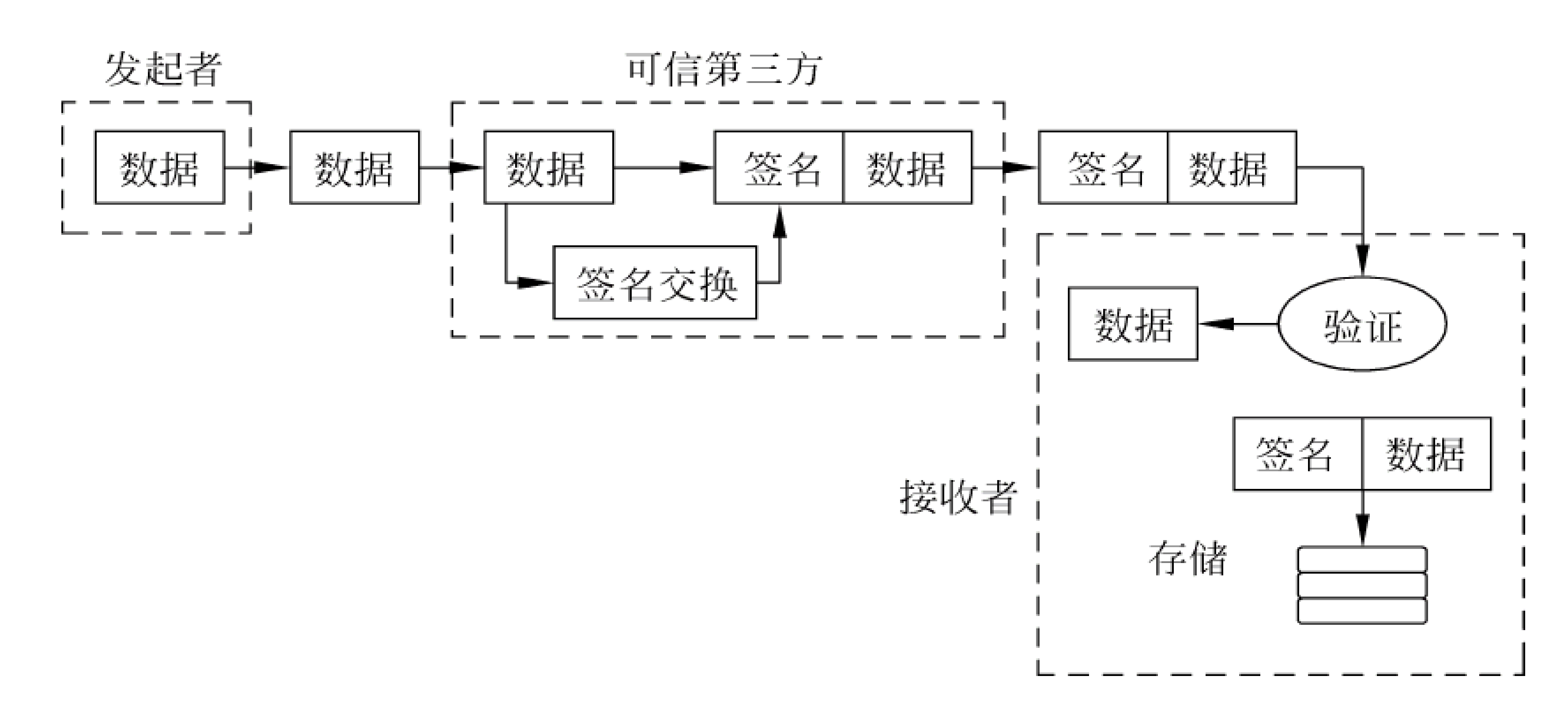


图 2-17 内联可信第三方——证据转发

5. 机制结合

所有基于数字签名的机制都可以增强到实现多方签名。这需要使用每一个独立签名或者一个多方签名,例如多方贡献出密钥的一部分以产生一个签名。这些方案大大地提高了对危及密钥的攻击的防御能力,还减少了对单方的依赖。

此外,上述机制可以结合起来。例如被产生的证据可以包括发起者的签名加上一个或多个可信第三方的签名,这些签名用不同的系统产生。

6. 时间戳

在很多应用中,安全时间戳为数据项产生,此外可以安全地把数据项和发起者联系起来。时间戳的使用是为了某些目的,如确定电子商务文件的归档时间。此外,安全时间戳可以作为一些非否认机制的基本要求。例如,在一个简单的发起者数字签名机制中,可能会发生由于发起者故意公开密钥然后鼓动撤销密钥从而导致自己以前的签名无效。为了对付这种攻击,就需要有联合签名和撤回请求的安全时间戳。

时间戳由时间戳服务生成,这是可信第三方认可的,可以提供可信的时间源。给定的数据项的时间戳可通过时间戳服务添加时间值到数据项或其摘要中,然后用数字签署结果获得。被处理的签名作为另一个非否认证据。时间戳功能可和另一个非否认签名功能结合,如一个可信第三方可以同时证明发起者身份和准确时间。作为一种选择,一个专用的时间戳服务可以应用到加另一个签名到证据上的情况。

2.5.3 传递的非否认

传递的非否认关系到关于是否某一特定方接受到一个特定数据项的潜在的争执和(或)关于传递发生时间的争执。注意这种形式的非否认没有超越传递,这里没必要暗含接收者阅读数据或在上面做了手脚。

在这种形式的非否认中,证据产生者这一角色包括了接收者,某些情况下还包括一个或

多个可信第三方。服务请求者包括数据产生和(或)代表了产生者利益的那些实体。证据验证者包括发起者和(或)可信第三方。

证据需要能无可辩驳地将各种信息片段联系或连接起来,至少包括:①接收者的身份;②数据项的精确值。某些情况下,还有其他信息,例如:①传递发生的日期和时间;②发起者的身份;③涉及证据产生的所有可信第三方的身份。提供传递的非否认的方法主要有以下几种。

1. 接收者签名确认

一个机制涉及接收者产生一个确认信息送回给发起者。这个信息是包含了以下某些信息的一个数字签名:①被传递的数据项的副本或内容摘要;②其他需要的证据,如传递时间。

签名必须由接收者或担保接收者的可信第三方生成。图 2-18 描述了这个机制应用到接收者数字签名中的操作。

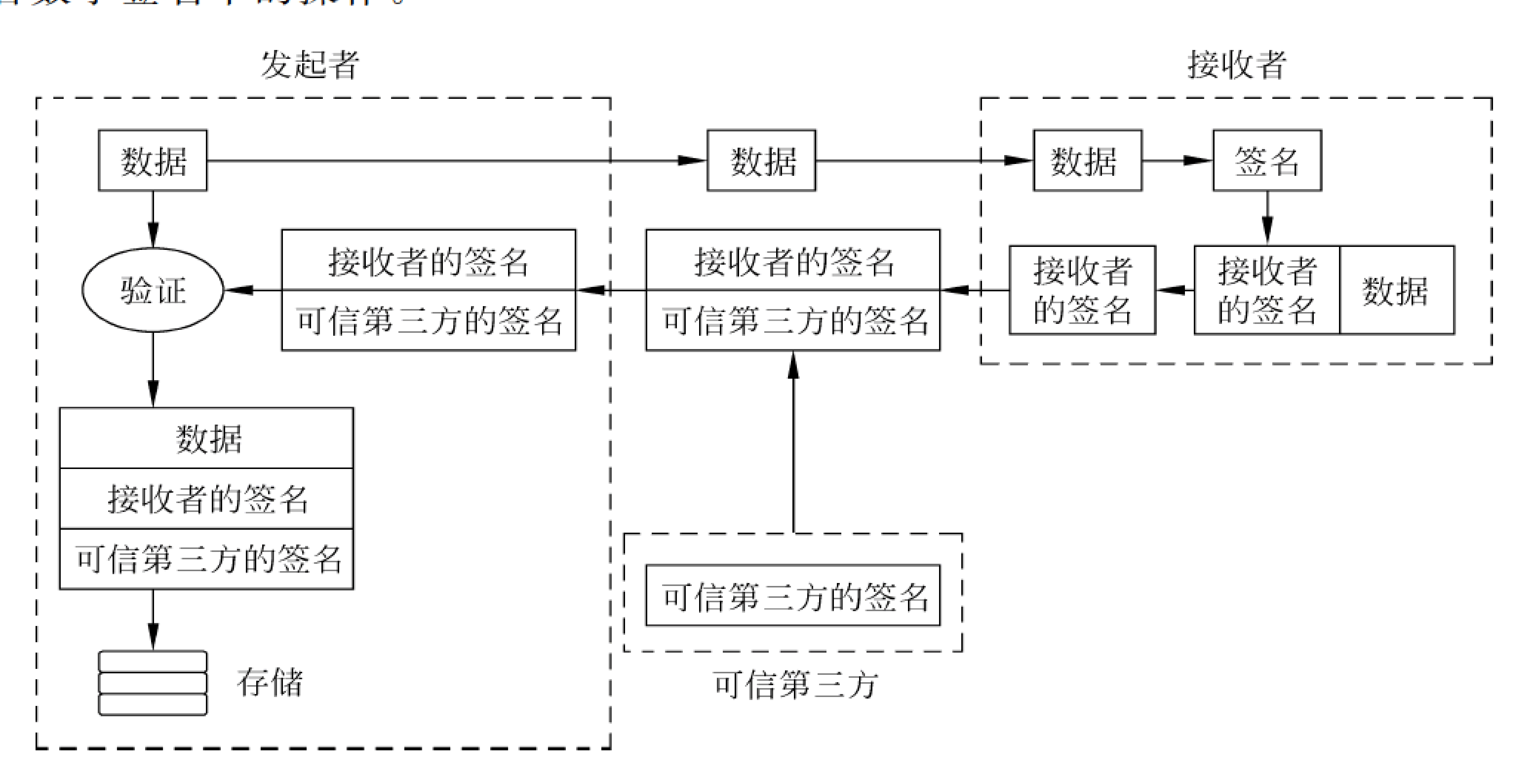


图 2-18　使用接收者的签名的传递非否认

在大多数情况下,这个机制等同于应用起源的非否认到包含必需证据的确认信息(除了它不需要真的把源信息或其摘要送回发起者)。这个传递的非否认机制中有关证书和密钥撤销的条件起源的非否认中的相关讨论一样。

2. 接收者令牌确认

作为上面方案的一个变体,一个由可信第三方用对称密码技术生成的令牌可用于替换签名。像起源的非否认方案——可信第三方令牌方案一样,令牌的验证(在这里由发起者发行)需要把它送回可信第三方。

3. 可信传递代理

上面的机制存在的一个问题是没有防止接收者在看了被传递数据内容后而不去确认的保护措施。可通过引进一个可信第三方作为传递代理来提供这样一种基本的保护措施。传递代理介入发起者和接收者的通信线路中,与接收者相当的近。发起者发送数据项到传递代理处,由它用一个类似于前面所说的基于签名的机制来确认收到的东西。通常,传递代理

直到最后传递到接收者才生成确认。然而,在某些情况下,传递代理先于最后到达的传递生成确认;这种情况下传递代理假定有责任确认数据项被传递到接收者。这种可信传递代理的方法类似于使用一个警官送某些合法文件给一个人。

4. 两阶段传递

两阶段传递是针对难对付的接收者问题特别设计的。它可用于当一个数据项从发起者到接收者或从可信代理传递到接收者的时候,这里不妨假设它是从发起者到接收者。发起者首先送一个预接收消息到接收者,该消息包括数据项的摘要但不是用接收者可以解释的形式的数据项。接收者用一个读接收消息回答,该消息包括一个摘要的签名副本。在同真正的数字签名比较从而检验该消息中的数字签名后,发起者传送整个数据项。接收者用进一步签名过的确认来回答,与用于基本的接收者的签名机制相同。

与基本的接收者签名机制相比较,两阶段传递机制的优点是在传送读取接收消息的过程中,接收者不仅确认当前通信线路的运行状态,而且回复关于含有被同意的摘要的数据项的最后收据,尽管完整的数据项还不知道。虽然可能会出现在预接收消息送出后马上出现通信失败,使得完整的数据项的传递耽搁,但发起者和接收者平等地负责数据项的传递。在没有预先交换的情况下,接收者会不顾已收到的特定数据项,假装发生不确定的通信失败。

如果传递时间的协调是重要的,也可以使用两阶段传递程序。接收者在那个签署过的读接收消息中包括了他当前的时间值。发起方可以检查该值是否合理,可以在传送完整数据项之前记录下来作为证据。

5. 渐进的传递报告

很多分布式通信应用程序,如电子邮件和EDI,涉及经过很多不同的管理区域转发数据项。那些应用程序要冒中途丢失数据项的风险,而且不能确认丢失在哪个特定的某方或区域。因此,可以协定当消息经过它的路径时应用多点传递的非否认,如在进入一个不同的管理区域的每个入口点处。

2.6 可　用　性

可用性使已经授权的用户(或计算机系统)可以在不受干涉和阻碍的情况下访问信息,并按所需的格式接受它。可用性要求无论何时,只要用户需要,信息系统必须是可用的,也就是说信息系统不能拒绝服务。网络最基本的功能是向用户提供所需的信息和通信服务,而用户的通信是随机的、多方面的(语音、数据、文字和图像等),有时还要求时效性。网络必须随时满足用户通信的要求。攻击者通常采用占用资源的手段阻碍授权者的工作。可以使用访问控制机制,阻止非授权用户进入网络,从而保证网络系统的可用性。增强可用性还包括如何有效地避免因各种灾害(战争、地震等)造成的系统损失。

Aura与Nikander建议使用无状态连接(stateless connection)抵抗连接损耗(connection depletion)攻击。他们让客户端存储服务器要求的所有状态信息,并且在每次消息发送时必须返回给服务器。此时,服务器不需要存储任何状态信息。但是返回给服务器的状态信息必须被服务器验证是真实的,而且状态信息必须保密。因此,通信协议(提供无状态连接)会

产生一个包含通信和计算量的总开销。一个实际的在服务器的端延迟状态需要的机制是cookies 的使用。

为了抵抗连接损耗攻击，Meadows 建议协议中的每个消息必须认证。然而，为了尽可能减少计算量，协议开始时认证比较弱，随着交换消息的增加，认证强度增大。Cookies 是由服务器端形成的，必须通过客户端返回，形成的认证可能较弱。Meadows 还在 Gong 和 Syverson 的失败-停止协议的基础上，提出了一个形式化的框架。

Juels 和 Brainard 给出了一个机制来提供较强的认证（强于 cookies 提供的认证），称为客户难题（client puzzles）。他们认为，当服务器端的负载变高时，服务器端将发送一个中等计算难度的"puzzle"给每个用户，这个 puzzle 必须在一个新的连接形成之前得到解决。这个要求不会对真实用户造成很大的困扰，但是对于试图建立多次连接的攻击者而言，将需要解决很多难题（puzzles）。

可用性服务即按照授权实体的要求有可被访问和可被使用的性质，也叫做服务保障。为了确保数据的可用性，系统必须同时利用防御机制和恢复机制。网络环境下的可用性不仅包括用户可访问用户代理和服务器等硬件和软件资源，还包括用户有能力获得所期望的使用具有合理吞吐量的网络带宽等服务质量（QoS）。网络流量必须能穿过局域和广域网按要求到达预期目的地。应用程序和数据的可用性与信息系统的安全密切相关。不可访问的数据就等于不存在，不能工作的应用程序也是毫无用处的。

保证可用性的最有效的方法之一是提供一个具有通用安全服务的安全网络环境。操作系统和应用程序应该具有一定的抗故障能力。通过使用访问控制阻止未授权资源访问，并利用完整性和保密性服务避免数据泄露或被修改，并可以防止可用性攻击。访问控制、完整性和保密性成为协助支持可用性安全服务的机制。

可用性问题的解决方案包括以下四种。

（1）避免受到攻击——一些基于信息系统的攻击旨在破坏、降级或摧毁网络资源。解决办法是加强这些资源的安全使其不受攻击。免受攻击的方法包括：关闭操作系统和网络配置中的安全漏洞；控制授权实体对资源的访问；限制对手操作或浏览流经或流向这些资源的数据从而防止插入病毒等有害数据；防止路由表等敏感网络数据的泄露。

（2）避免未授权使用——当资源被使用、占用，或过载时，其可用性会受到限制。如果未授权用户占用了有限的处理能力、网络带宽、调制解调器连接等资源，则这一资源对授权用户就是不可用的。对这些资源使用者的标识和鉴别可以提供访问控制来限制未授权使用。然而，过度频繁地发出标识和鉴别请求可能导致网络运行减慢或停止。

（3）防止例程（rutine）失败——正常的操作失误和自然行为也可能导致系统可用性降低。解决方法包括使用具有高可靠性的设备、提供设备冗余和提供多路径的网络连接。

（4）加强冗余和数据恢复机制——包括旁路信道、数据备份通道和实时热备等能力的信息存储和流动的冗余是必需的，以绕过任何有问题的区域，并保证数据不会因为密钥或者存储介质等的损坏而无法使用。

可信操作系统的思想也可用于限制信息系统攻击的有害影响。通过抑制恶意代码造成的损失和保证其他安全机制的适当操作，可信操作系统便能保证可用性。可信操作系统显现的另一特征是操作的完整性。它确保在终端系统中执行的进程能够提供一致的、可重复的结果，而且该结果不受未授权访问的影响。

关键的系统组件必须提供物理安全保证，不仅要避免遭到攻击和资源滥用，还要保证平台和应用程序不被修改，因为通过某些修改可以绕过那些提供可用性的安全服务调用。

2.7 安全审计和报警

安全审计和报警服务，与其他安全服务的不同之处在于没有单个的特定安全机制可以直接用来提供这种服务。审计机制也许可以特征化为基于大量管理和操作方法的处理过程。

2.7.1 安全审计

系统安全审计是指对安全活动进行识别、记录、存储和分析，以查证是否发生安全事件的一种信息安全技术。安全审计为安全管理人员提供可进行分析的管理数据，以发现在何处发生了违反安全方案的事件。利用安全审计结果可以调整安全政策修补系统漏洞。

安全审计是对系统记录和活动的独立评估、考核，它测试系统控制的充分性，确保与既定策略和操作规程相一致，有助于对入侵进行评估，并指出控制、策略和程序的变化。安全审计跟踪是一种很有价值的安全管理机制，可通过事后的安全审计来检测和调查安全策略执行的情况以及安全遭到的破坏情况。

安全审计应该具备以下功能：

（1）记录关键事件。由安全管理人员定义违反安全策略的事件，并决定将什么信息记入审计日志。

（2）提供可集中处理审计日志的数据形式。以标准的、可使用的格式输出安全审计信息，使安全管理人员能够直接利用软件工具处理这些事件。

（3）提供易于使用的软件工具。为了能立即发现违反安全规定的事件，要为安全管理人员提供一套易于使用的基本分析工具。

（4）实时安全报警。扩展现有管理工作的能力，并将它们与数据链路驱动程序和本地审计能力结合起来，当发生与安全有关的事件时，安全管理人员就会接到报警。

可以看出，安全审计的主要功能是记录和跟踪信息系统状态的变化以及安全相关的事件，如用户的活动等，并对程序和文件的使用情况进行监控，记录对程序和文件的使用以及对文件的处理过程。

安全审计是系统记录和活动的独立检查和验证。安全审计的目的包括：

（1）辅助辨识和分析未经授权的活动或攻击；

（2）帮助保证和相应实体对非授权活动进行相应处理；

（3）促进开发改进的损伤控制处理程序；

（4）评估与已建立的安全策略的一致性；

（5）报告那些可能与系统控制不相适应的信息；

（6）辨识可能需要的对控制、策略和处理程序的改变。

审计日志是记录信息系统安全状态和问题的依据，日志管理是确保记录长期稳定和有用的过程。

1）日志的内容

基于安全观点考虑，理想的日志应该包括全部与数据、程序以及系统资源相关事件的记录。实际上，这样的日志只能适用于某些有特殊需要的系统，因为它所付出的代价太大，所以，最好根据系统的安全目标和操作环境单独设计日志。

在决定日志记录什么内容时，要充分考虑安全需求，需求应该以明确的条目加以说明，包括何种类型的数据需要保护，系统怎样识别这些数据，各类数据准确程度的必要性，谁可以存取保护数据以及系统如何识别授权用户，等等。

日志作为对系统实行强制性安全措施的综合研究对象，内容应该包括记录的信息类型、应用的参数、所做的分析、产生的报告和数据的保留期。

2）日志的作用

当存在非法使用系统的行为时，日志可以为调查处理工作提供有效的证明。日志还可以作为责任认定的依据，当发生责任纠纷时，查阅日志不失为一种好的方法。另外，日志作为系统运行记录的集合，对分析系统运行情况、排除故障和提高效率都会起到很好的帮助作用。

3）日志的管理

日志管理最典型的方法是日志转储，即将旧的、已写满的日志文件转存到另一边，用新的空日志文件占用它们的位置。正确转储日志以后，还必须注意备份，并提供查询和检索功能。

2.7.2　安全报警

鉴别、访问控制、机密性和完整性等这些安全服务都是对信息系统的被动防御。但是，不能担保这些服务总能正常运行。由于不合适的保护机制或保护机制中的故障，可能会产生新的攻击或这些安全机制本身不能解决的情况（例如密码被盗），所以系统总会有安全泄密的风险。因此，需要有一个方法能够检测出安全的泄露或有可疑的事件发生，并将这些情况报告给操作员或管理员。安全报警是指用户或进程发出的警告，以指示发生了情况，可能需要及时采取行动。安全报警的目的包括：

（1）报告实际的或明显的违背安全的企图；

（2）报告各种与安全相关的事件；

（3）报告到达一定门限（阈值）后触发产生的事件。

一个与安全相关的事件会触发一个安全警报，从原理上讲，任何网络或系统部件都能够检测出该事件。在管理模型中，检测事件的部件是受管对象。安全报警通过 M-EVENT-REPORT 通知给管理系统，如图 2-19 所示。

安全报警报告功能标准（ISO/IEC 10164-1）描述了 M-EVENT-REPORT 调用中所传递的信息。受管信息定义（ISO/IEC 101652）中详细说明了交换中所使用的正确抽象语法。

安全报警报告中传递的参数分为三类：

（1）在 ISO/IEC 9595 中定义了所有的 M-EVENT-REPORT 中通用的参数，包括调用标识符、模式、受管对象类、受管对象事例、事件类型、事件时间和当前时间等。

（2）在 ISO/IEC 10164-4 中定义了所有管理警报通用的参数，包括通知标识符、相关的

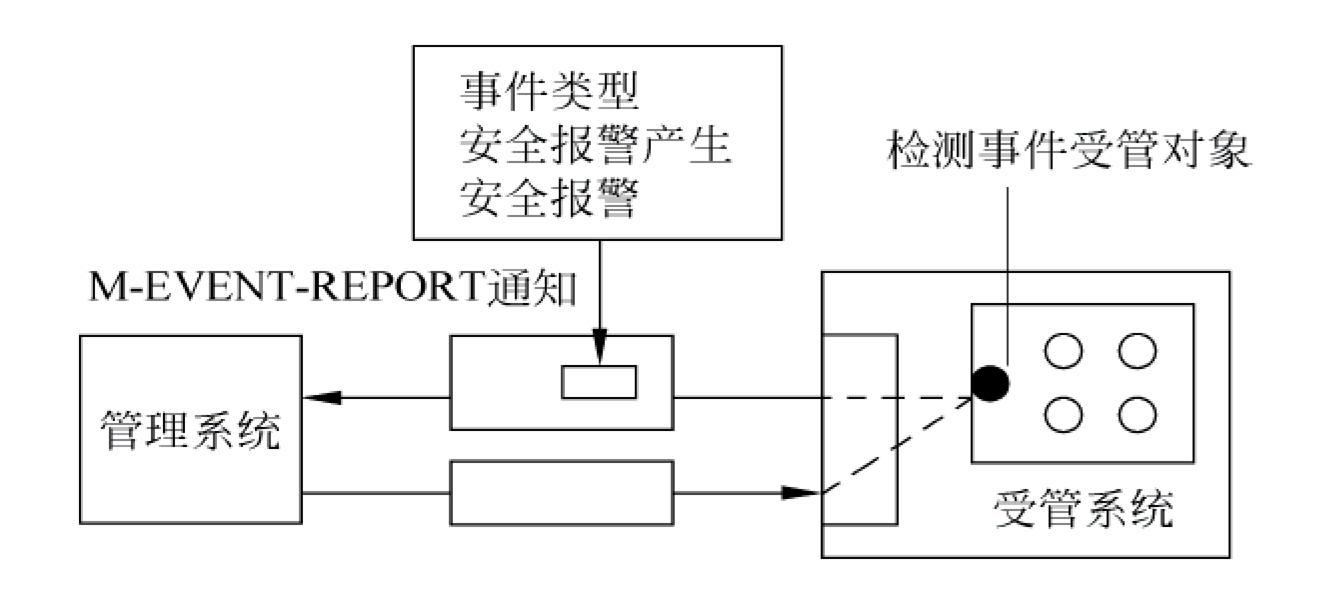


图 2-19 安全警报报告过程

通知、额外的信息和额外的文本等。

(3) 安全报警特有的参数，包括安全报警原因、安全警报的严重性、安全报警检测器、使用服务的用户和服务的提供者等。

事件类型和安全报警原因的组合表明了警报的原因，其可能的组合有以下几种。

(1) 完整性破坏：指出未授权的修改、插入或删除数据的事件。安全报警原因的可能值是：复制信息、信息的丢失、信息修改的检测、顺序混乱的信息和不希望的信息。

(2) 违规操作：指明不能获得的信息、违法行为或一些服务的不正确调用事件。安全报警原因的可能值是：拒绝服务(故意阻止服务的合法使用)、超出服务、过程出错和未陈述原因。

(3) 物理侵入：指明对物理资源有可疑攻击的事件。安全报警原因的可能值是：损害电缆(对通信介质的物理破坏)、入侵检测(非法进入某个站点或损害设备)和未陈述原因。

(4) 安全服务或机制的侵犯：指明一个安全服务机制或机制检测到潜在的攻击事件。安全报警原因的可能值是：鉴别失败、破坏机密性、抗抵赖的失败、未授权的访问企图和未陈述原因。

(5) 时间区域的侵犯：指明在不希望的或禁止的时间里发生某些事的事件。安全报警原因的可能值是：延迟的信息(接到信息的时间比预定时间要晚)、密钥过期(使用过期的密钥)和上班时间外的活动(在不希望的时间里使用资源)。

安全报警的安全参数指明了由初始受管客体发觉的警报的意义，可能值有以下几个。

(1) 不确定：系统的完整性是未知的。

(2) 危险：安全性被损害危及到系统的安全。系统可能不能再正常运转来支持安全策略。例如，未授权的修改或泄漏与安全有关的敏感信息，如系统密码或违反物理安全等。

(3) 主要：检测到违反安全，并且重要的信息或机制已经遭到损害。

(4) 次要：检测到违反安全，并且不太重要的信息或机制已经遭到损害。

(5) 警告：不相信系统的安全性受到威胁。

安全报警检测器参数是标识检测警报条件的实体；使用服务的用户参数标识那些请求服务从而导致警报发生的实体；服务提供者参数标识那些提供服务从而导致警报发生的实体。

习　题　2

1. 常见的认证技术有哪些？
2. 访问控制列表的基本特征是什么？
3. 简单描述一下基本的访问控制策略的方案。
4. 什么是 Bell-Lapadula 安全模型和 Biba 安全模型？
5. 如何实现机密性保护？如何实现完整性保护？
6. 简单描述一下不可否认服务的实现过程。
7. 安全审计有哪些功能？
8. 说明安全报警的报告过程。

第3章　操作系统安全

操作系统是提供对各种硬件资源访问权限的软件，没有安全的操作系统就没有安全的信息系统，操作系统安全是信息系统安全的基础。

3.1　保护对象和保护方法

多程序的出现意味着计算机系统的内存、输入/输出设备（如硬盘、打印机和磁带装置）、共享程序和子程序、网络、共享的数据等都需要被保护。操作系统使用一些安全机制来加强对这些对象的保护，许多操作系统的保护机制由硬件来提供。

保护的基础是分离，使得一个用户的对象与其他用户的对象相分离。操作系统中的分离可以通过下面这些方法实现：

（1）物理分离，此时不同的用户使用不同的物理对象，比如根据输出数据的安全等级的不同分离打印机，这样提供了很强形式的分离，但不现实。

（2）时间分离，要求不同的进程在不同时间执行，避免了并发带来的许多问题。

（3）逻辑分离，逻辑分离通过沙箱实现，每个进程在各自的沙箱内运行，进程在它的沙箱内可做任意的事情，但限制它在沙箱外的行为。沙箱是一种按照安全策略限制程序行为的执行环境。早期主要用于测试可疑软件等，比如黑客们为了试用某种病毒或者不安全产品，往往可以将它们在沙箱环境中运行。

（4）加密分离，此时进程加密他们的数据和计算，使得外部进程很难理解它们。

当然，分离形式的相互组合也是可能的。

分离的种类是按照实现复杂性递增的顺序列出的，前三个又是按所提供的安全性的递减顺序排列的。尽管如此，前两种方法很严格，且导致资源利用率不高。因此，希望将保护的负担转移给操作系统，从而允许安全需求不同的进程并发执行。分离只解决了一半问题。我们想分离用户和他们的对象，但是也希望那些对象中的一部分提供共享。比如，两个不同安全等级的用户想要调用相同的搜索算法或函数。同时希望这两个用户能够共享算法和函数而不放弃他们自己的安全需求。操作系统能够以多种方式支持分离和共享，提供不同安全级别的安全。

（1）无保护。当敏感程序不同时运行时，操作系统不提供保护也是可以的。

（2）隔离。当操作系统提供隔离时，同时运行的不同进程不会感觉到彼此的存在。每个进程有它自己的地址空间、文件和其他对象。操作系统必须以某种方式限制每个进程，这样每个进程的对象是完全被隐藏的。

（3）共享所有或无共享。如果是这种形式的保护，那么对象的所有者宣布它是公开的还是私有的。一个公开的对象对所有的用户可用，相反一个私有的对象仅对它的所有者可用。

(4) 基于访问限制的共享。使用访问限制,操作系统检查每个用户对一个对象的访问权,即对具体的用户和对象实施访问控制。操作系统决定一个特殊的用户是否能够访问一个特殊的对象。

(5) 基于能力的共享。基于能力的共享是对基于访问限制的共享的扩展。这种形式的保护允许对象共享权限的动态创建。共享的程度取决于主体的所有者,计算的上下文,或者对象自身。

(6) 对象的限制使用。这种形式的保护不仅限制对象的访问权限,还限制对象被访问后的使用。比如,用户可能被允许查看一个敏感文件,但是不允许打印这个文件的复本。更有甚者,可能允许用户访问数据库中的数据,从而得到统计摘要(比如某一个特定阶层的平均收入),但是不能获知具体的数据值(每个人的收入)。

这六种共享模式是按照实现的难易程度的递增顺序排列的,它们提供的保护的程度逐渐增强。一个操作系统可能对不同的对象、用户或者状态提供不同级别的保护。

对于数据而言,访问权限可以控制在不同等级:位、字节、元素或字、域、记录、文件集。这样,就出现了控制粒度的概念。对象控制的等级越高,访问控制的实施越容易。然而,有时操作系统必须允许的访问远多于用户的需要。比如,对于大的对象,必须授权给仅需要访问对象的一部分的用户(如文件中的单个记录)整个对象(整个文件)的访问权限。

3.2　内存与地址保护

3.2.1　电子篱笆

多程序最明显的问题是阻止一个程序影响其他用户内存空间中的数据和程序。

内存保护的最简单的形式是单个用户操作系统中防止一个错误的用户程序破坏操作系统的固有部分。正如它的名字,篱笆是限制用户到边界的一边的方法。

执行过程中,电子篱笆是预先规定的内存地址,它使得操作系统在一边,用户在另一边,图 3-1 是这种情况的一个例子。不幸的是,这种执行很严格,因为预先规定的空间值总是为操作系统保留的而不管操作系统是否需要。如果要求的比预先规定的空间小,那么多余的空间就会被浪费。相反,如果操作系统需要更多的空间,那么它的增加不能超过篱笆的界限。

另一种实现方案是使用硬件寄存器,通常称为篱笆寄存器(fence register),包含操作系统的终端地址。与固定的篱笆相比,这个方案中篱笆的位置能够修改。每次一个用户程序为数据修改产生一个地址,这个地址自动与篱笆寄存器的内容相比较。如果这个地址大于篱笆寄存器地址(即用户区域),那么执行指令;如果小于篱笆寄存器地址(即操作系统的区域),那么出现错误状态。篱笆寄存器的使用如图 3-2 所示。

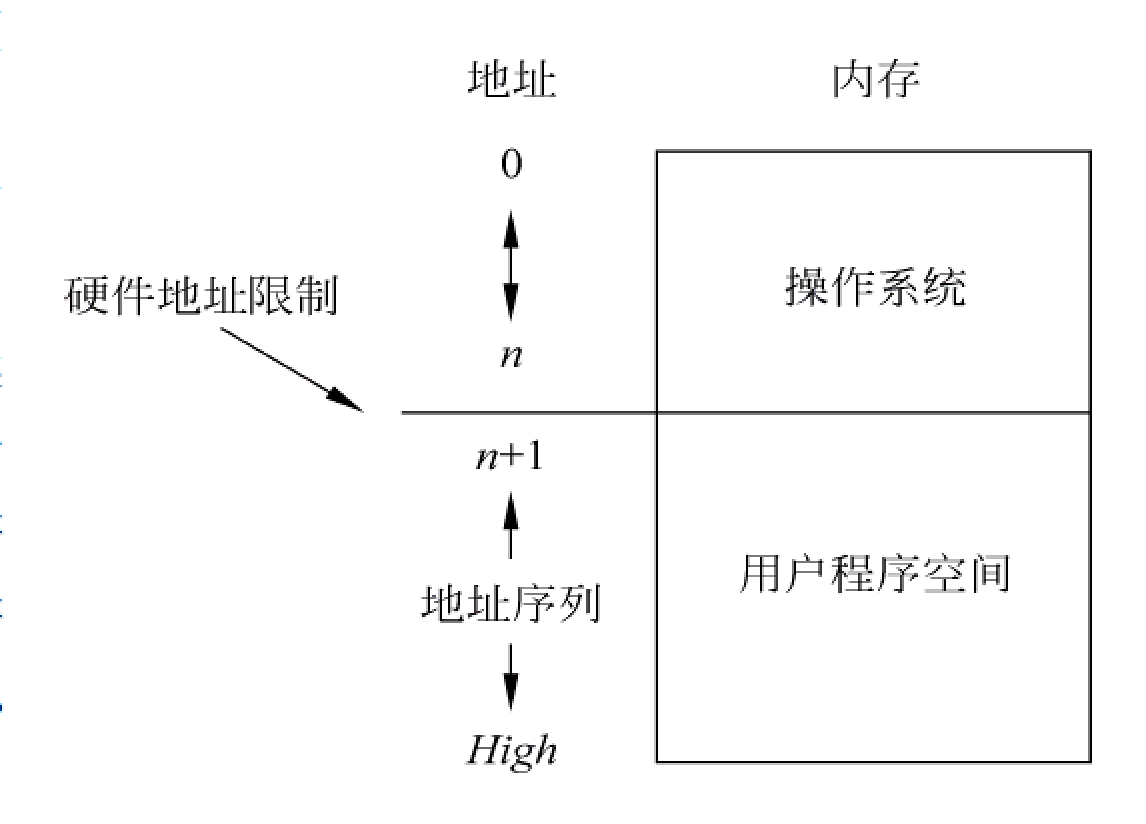


图 3-1　固定的篱笆

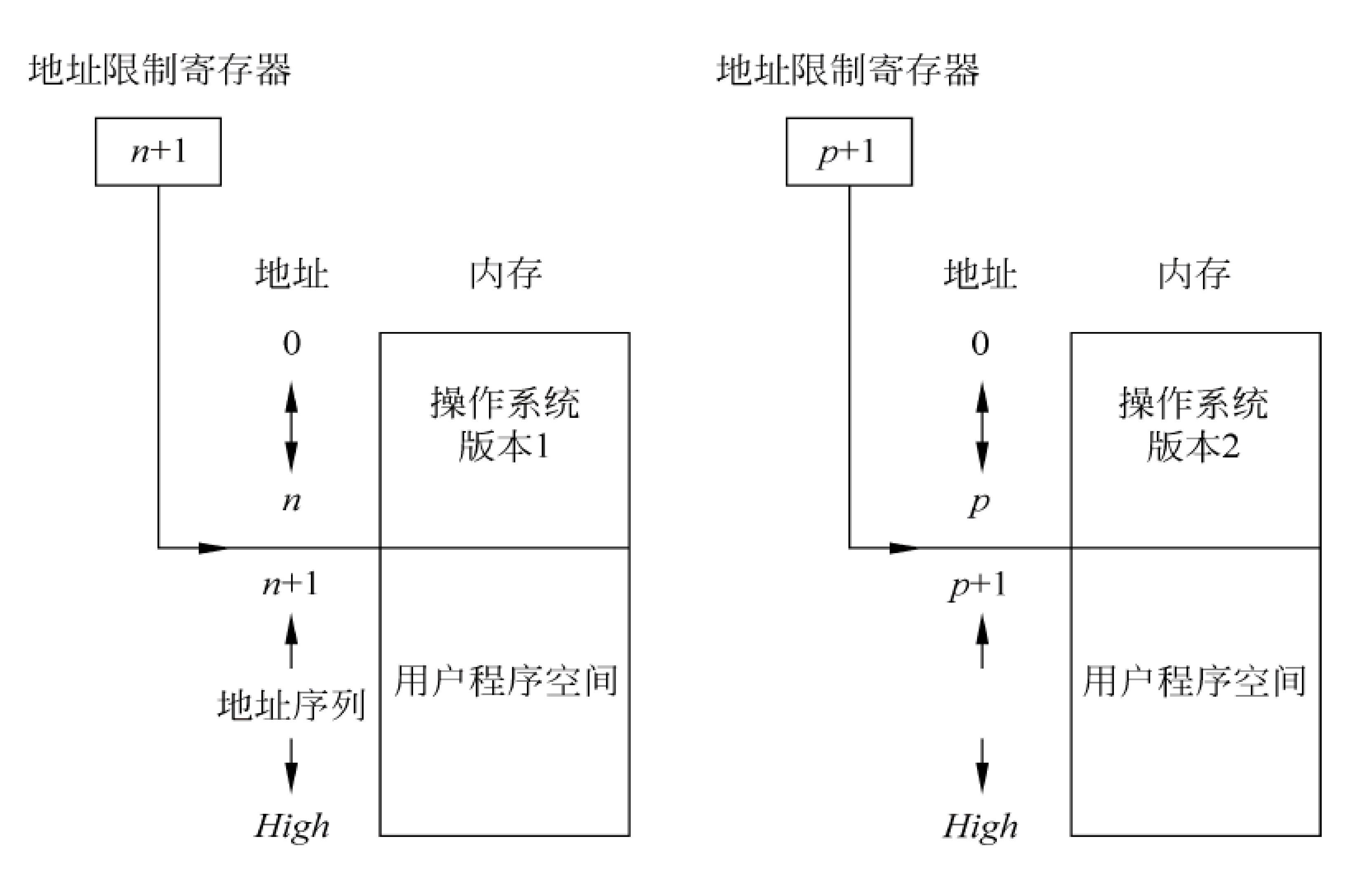


图 3-2 可变的篱笆寄存器

篱笆寄存器的保护仅是一个方面。换句话说,篱笆寄存器能保护操作系统不受用户程序的侵扰,但不能保护一个用户程序不被另一个用户程序侵扰。相似地,用户不能鉴别程序的某个区域能否被访问(如程序本身的代码或只读的数据区域)。

3.2.2 重定位

如果能够使得一个操作系统有固定的大小,那么程序员便可以写入他们的代码,这里假设程序开始于一个常量地址。操作系统的这个特征使得它很容易确定程序中任何对象的地址。然而,这也使得不可能改变初始地址,比如,操作系统的一个新版本比旧版本大或小。如果操作系统的大小允许改变,那么程序必须以一种方式写入,且这种方式不依赖于内存中具体地址的分配。

重定位是将一个程序从地址 0 开始写入的过程,它改变所有的地址来反映实际的地址(程序在内存中的地址)。在许多例子中,这样做仅仅是为程序的每个地址增加了一个常量重定位因子,即重定位因子是分配给程序的内存的开始地址。

一般地,在这种情形下使用篱笆寄存器可以有更多的好处:篱笆寄存器可以是一个硬件重定位设备。将栅栏寄存器的内容添加到每个程序地址。这个过程不仅可以实现重定位功能,也保证了没有人能够访问比计数器值更小的地址。这是因为地址是一个无符号整数,对任何数增加篱笆寄存器的值都能够保证产生的结果等于或高于篱笆地址。当合法程序访问操作系统的某个地址时,相应的指令会被添加进去。

3.2.3 基/界寄存器

使用篱笆寄存器的操作系统的主要优点是可以进行重定位,这个特征在多用户环境中显得尤其重要。对于两个或更多的用户,没有人能够事先知道执行程序在哪里被载入。重定位寄存器解决了这个问题,它提供一个基(或开始)地址,所有的程序内部地址与基地址抵消。可变篱笆寄存器一般也称为基寄存器(base register)。

篱笆寄存器提供下界(起始地址)但不提供上界。上界对于知道空间大小和检查溢出

(进入“禁止”区域)方面很有用。为了克服这个困难,常常添加第二个寄存器,如图 3-3 所示。这第二个寄存器,也称为界寄存器(bounds register),是地址限制的上界,同理,基或篱笆寄存器是地址限制的下界。每个程序地址必须在基地址之上,因为基寄存器的内容被加入了地址;检查每个地址保证它们在界地址内。通过这种方式,程序的地址空间在基寄存器和界寄存器之间。

这个技术防止程序的地址被其他用户修改。当执行的程序的主体从一个用户变为另一个用户时,操作系统必须改变基寄存器和界寄存器的内容,从而反映给用户真实的地址空间。这种转换称为上下文转换(context switch),当控制从一个用户转换成另一个用户时,操作系统必须执行上下文转换。

使用基/界寄存器,用户能够很好地与外部用户分离,更准确地说,可以防止外部用户受到其他用户程序中错误的干扰。用户地址空间中错误的地址仍然影响程序,因为基/界检查的仅仅是每个地址是否在用户地址空间内。比如,当用户空间的下标超出范围或未定义变量在用户地址空间内产生一个地址引用时,用户错误就可能出现,但是,不幸的是,错误发生在用户程序的执行指令内部。这时,用户可能碰巧存储的是指令之上的数据。这样的错误(用户一般不会注意到)会破坏程序,但仅仅是破坏用户自己的程序,不涉及其他用户。

要解决这一问题,可以再使用一对基/界寄存器对重写程序。一对用于程序的指令(代码),重定位和检查取得指令(将被执行的指令);另一对用于数据空间,重定位和检查数据访问(指令的操作)。两对基/界寄存器的使用如图 3-4 所示。尽管两对寄存器并不能防止所有的程序错误,但是它们限制了数据操作地址空间的指令产生的影响。两对寄存器的使用还有一个重要的优点,即一个程序分离后得到的两部分可以各自进行重定位。

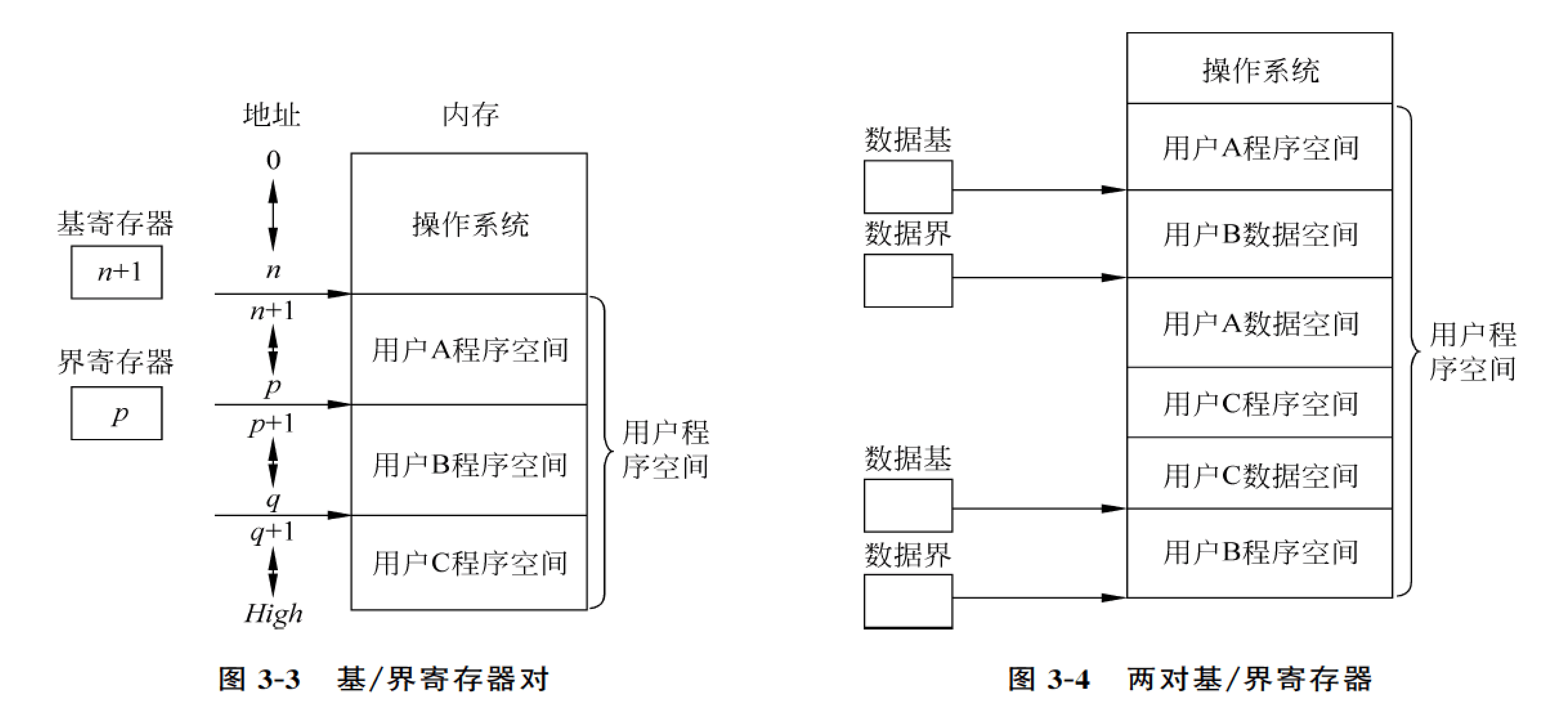


图 3-3　基/界寄存器对

图 3-4　两对基/界寄存器

寄存器还可以扩展成三对或更多对寄存器:一对用于代码,一对用于只读数据,一对用于可读写的数据值。尽管理论上说这个概念可以被延伸,但是实际的计算机设计只允许两对寄存器。对每对增加的寄存器(两个以内),每条指令的机器码必须隐含哪个重定位对用来处理操作指令。即使用两对以上的寄存器,每个指令都要对两个或更多的数据空间中的一个进行说明。但是仅使用两对,这个策略是自动的:指令使用一对,数据使用一对。

3.2.4 标记体系结构

使用基/界寄存器进行保护或重定位的另一个问题是它们的连续性。每对寄存器都限制对连续地址的访问。编译器或下载器很容易就能对程序进行重排列,这样所有的代码片段是相邻的,所有的数据片段也是相邻的。

然而,有时候我们可能只想保护某个数据而不是全部数据,比如,个人资料可能要求保护薪水情况,而办公地点和电话号码不需要保护。此外,程序员可能希望保证一定数据值的完整性,比如仅在程序初始化时允许这些数据被写而不允许程序以后修改它们,这样能够避免程序自身代码的错误。程序员也可能请求库文件中的一个共享的子程序。无论是操作系统问题还是运行的程序问题,使用好的设计方案都可以使得这些问题得到解决。

操作系统的一些特殊的设计特征也可以起作用。基/界寄存器为共享创建了一个 all-or-nothing 条件:程序或者允许它所有的数据被访问或修改,或者禁止所有的访问。即使有第三组用于共享数据的寄存器,所有的数据也需要放置在一起。一个进程不能用一个模块共享数据条目 A、B 和 C,用另一个模块共享 A、C 和 D,而用第三个模块共享 A、B 和 D。完成这种共享的唯一方法是将每组合适的数据值移到一些相邻的空间中。不过,如果数据条目是大的记录、排列或结构,这个解决方案就不可行。

一个可行的方案是标记体系结构。在这个方案中,机器内存中的每个字都有一个或多个临时位来区分对该字的访问权限。这些临时位仅通过有特殊权限的(如操作系统)指令设置。每次指令访问时临时位都会被检测。

举一个例子,如图 3-5 所示(其中,R=Read-only,RW=Read/Write,X=Execute-only),一个内存地址的访问权限可能为仅执行(如指令的对象代码),另一个为仅取(如读)数据,还有一个为可修改(如写)。此时,两个相邻地址有不同的访问权限。使用一些临时的标记位,不同种类的数据(数字、符号、地址或者指针,以及无符号)能够被分离,使得数据域仅能被一些具有特权的进程(如操作系统)访问。

标记	存储字
R	0001
RW	0137
R	0099
X	…
X	…
X	…
X	…
X	…
X	…
X	…
R	4091
RW	0002

图 3-5 标记结构示例

这种保护技术已经用于一些系统中,尽管标记位的位数还相当少。Burroughs B6500-7500 系统使用三个标记位来分离数据字(三种类型)、描述符(指针)、控制字(堆栈指针和地址控制字)。IBM 系统/38 使用一个标记位来控制完整性和访问。

有一种方法是在一组连续的地址上使用一个标记位,比如 128 或 256 比特。每块地址区域一个标记位,执行时所增加的开销并不比每个区域一个标记位所需要的开销高。Intel I960 扩展体系结构处理器使用了一种标记体系结构,与一般的数据或指令的地址不同的是,它将标有“capability”的每个存储字占一位。“capability”控制对大小可变的存储块或片段的访问。这个标记值支持大小在 64~4 000 000 比特之间的存储片段,它有 2^{256} 个不同的保护域。

代码兼容性给标记体系结构提出了一个难题。就目前来看,标记体系结构可能并不比许多现代的方法更有效。事实上,大部分生产商仍然被锁定在传统的存储结构中,因为部件的广泛的可用性以及保持操作系统和机器系列的兼容性的要求。标记体系结构要求对所有

的操作系统代码从根本上进行改变,这样的代价是相当高的。但是随着存储器价格的降低,实现标记结构的可行性越来越高。

3.2.5　段式保护

段式保护即把程序分成单独的几个部分。每个部分都有一个逻辑统一体,显示它所有的代码和数据值之间的关系。比如,一个片段可以是一个过程的代码、一个数组的数据或者用于特定模块的本地数据值的集合的代码。分段可以产生与不限定数目的基/界寄存器相同的效果,换句话说,分段允许一个程序被分成许多访问权限不同的片段。

每个片段都有一个唯一的名字。片段中的代码或数据条目可以表示为＜名字,偏移量＞,这里名字是指包含数据条目的片段的名字,偏移量是指它在片段中的位置(即它与起始片段之间的距离)。

从逻辑上看,程序员描绘的程序是一个长的片段集合。片段可以各自进行重定位,它们可以被放置在任何可用的存储位置。逻辑片段和它真正的存储位置之间的关系如图 3-6 所示。

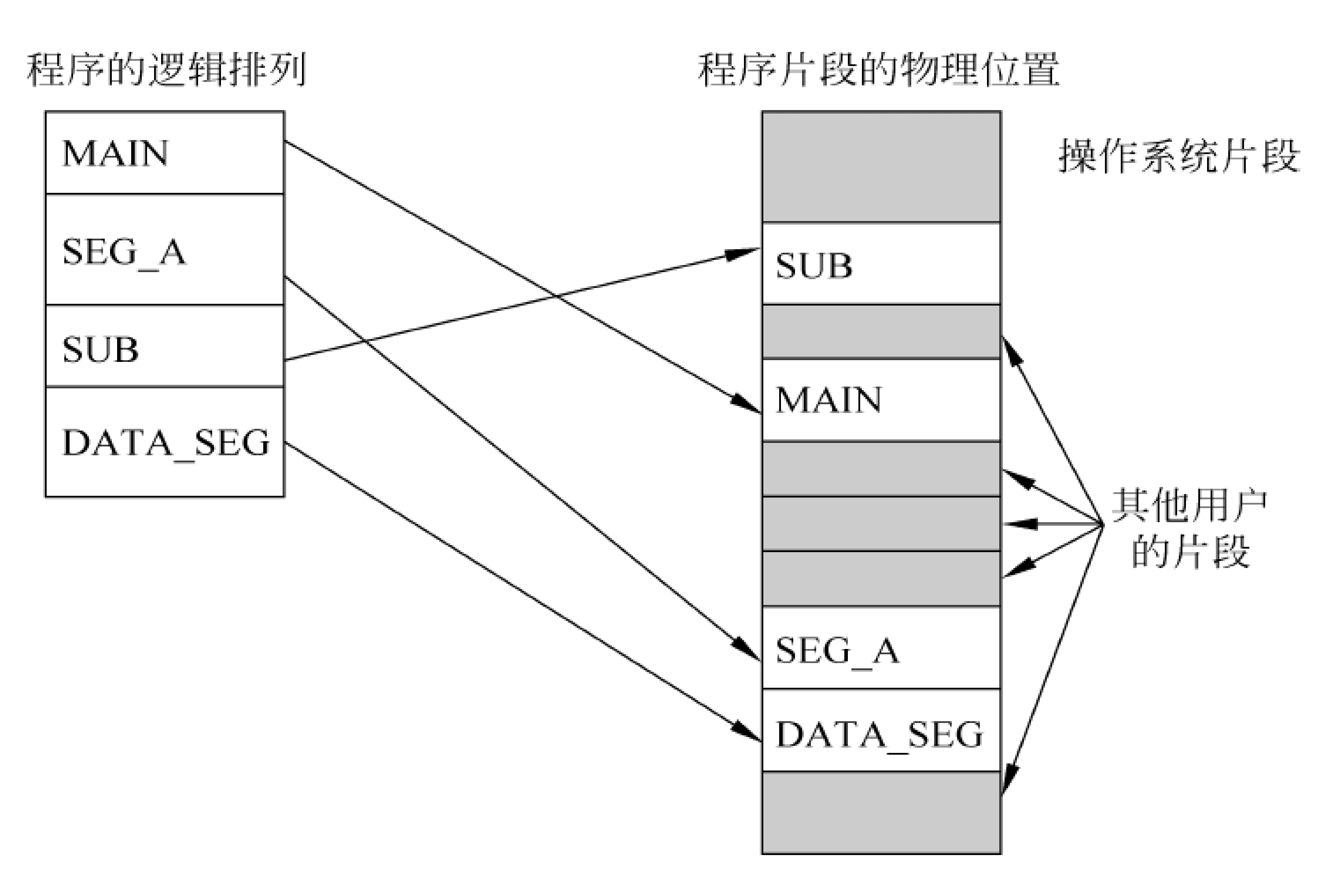


图 3-6　片段的逻辑表示和物理表示

操作系统必须维护一张表,该表包含片段名字和片段在内存中的真实的物理地址。当一个程序产生形如＜名字,偏移量＞的地址时,操作系统在片段目录中查找名字,确定它实际的起始存储地址,然后把偏移量加至地址上,这样就得到了数据代码的实际地址。这个转变过程如图 3-7 所示。考虑到效率问题,通常每个执行的进程都有一个操作系统的片段地址表。如果两个进程需要共同访问一个片段,那么它们的片段表中有相同的片段名字和地址。

这样,用户的程序不知道它使用的真正的物理地址是什么,因为没有方法能够确定与＜名字,偏移量＞相关联的真正的地址,当然也没必要知道,＜名字,偏移量＞对于程序访问它所需的任何指令和数据已经足够了。

这种隐藏地址用于操作系统有三个优点:

(1) 操作系统能够把任何片段放置到或移动到任何位置,即使程序已经开始执行。因

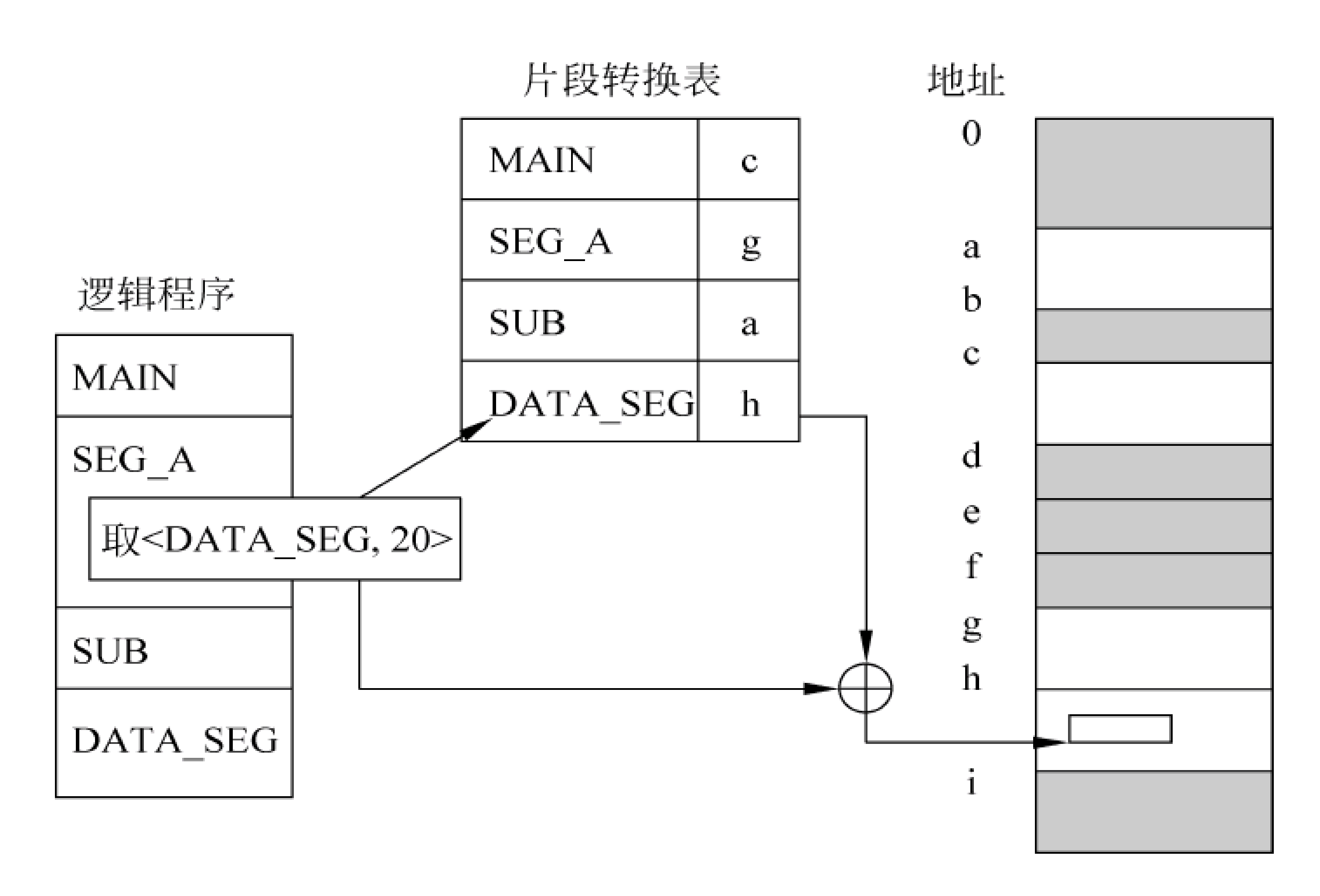


图 3-7 片段地址的转换

为它通过片段地址表转换了所有的地址引用,当片段移动时,操作系统仅需要在一个表中更新地址。

(2) 如果片段当前没有被使用,那么它可以从主存储中移出(存储到一个辅助设备中)。

(3) 每个地址引用都通过操作系统,所以可以检查每一个地址以保护操作系统或其他用户的程序。

因为第 3 个优点,所以只有当进程出现在程序的片段转换表中时,这个进程才能够访问片段。操作系统对哪个程序可以进入它们的片段地址表中的特殊片段进行控制。这种控制对片段提供强保护,阻止未授权的进程对它们进行访问。比如,程序 A 可能可以访问用户 X 的片段 BLUE 和 GREEN,但是不能访问 X 或其他用户的其他片段。通过一种简单的方法,可以允许一个用户对一个程序的不同片段有不同的保护等级。比如,第一个片段可能是只读数据,第二个片段可能是只执行代码,第三个片段可能是可写数据,这种情形下,分段能够估计出一个程序的不同部分的各自的保护目标。

分段机制提供了以下几点安全措施:①每个地址引用都被检查;②不同等级的数据条目可以分配不同级别的保护;③两个或更多的具有不同访问权限的用户可以共同访问一个片段;④对一个未授权的片段,用户不能产生对应的地址,也不能对其进行访问。

分段存在的问题是片段的大小问题。每个片段都有一定的大小,程序可以对有效的片段名产生一个引用,但是其偏移量超出分段的末端。比如,引用<A,9999>看起来是有效的,但是实际上片段 A 的长度可能仅为 200 字节。如果不采取措施,仅使用一个大的地址偏移量,这个安全漏洞就可能允许程序访问超过片段长度的任何内存地址。

这个问题在编译时不能被阻止,即使程序被加载时也不能,因为片段的有效使用要求它们在执行过程中可以增加大小,比如,一个片段可能包含一个堆栈之类的动态数据结构。因此,分段的安全实现要求检查一个生成的地址,以检验它有没有超过片段引用的当前长度。虽然这个检查结果需要另外的开销(时间和资源),但是分段系统必须执行这个检查;分段过程必须保存转换表中当前片段的长度,并比较每个生成的地址。

这样,为了平衡保护和效率,需要找到尽可能有效的方法进行分段。尽管如此,分段的有效实现带来了两个问题:第一,分段的名字在指令中编码不是十分方便;第二,操作系统

在转换表中查找名字可能不是很快。为了克服这些困难，片段名常常通过编译器转化成数字；编译器也附加一个链接表将数字和真正的片段名进行匹配。不幸的是，当两个程序需要共享一个片段时，这个方案的实现会有困难，因为分配给数据的片段数字必须与被片段访问的一致。

3.2.6　页式保护

程序被分成的大小相等的部分称为页，内存被分成的相等大小的单元称为页面（page frames），页的大小通常在 512～4096 字节之间。与段式保护相似，页式保护方案的每个地址也用一个二元组表示：＜页，偏移量＞。

每个地址的转换过程与分段相似：操作系统一张表，记录用户的页数和在内存中的真实地址。每个＜页，偏移量＞引用的页部分通过查表转化成页面地址，偏移量加入页面地址生成＜页，偏移量＞在内存中的的真实地址。这个过程可以如图 3-8 所示。

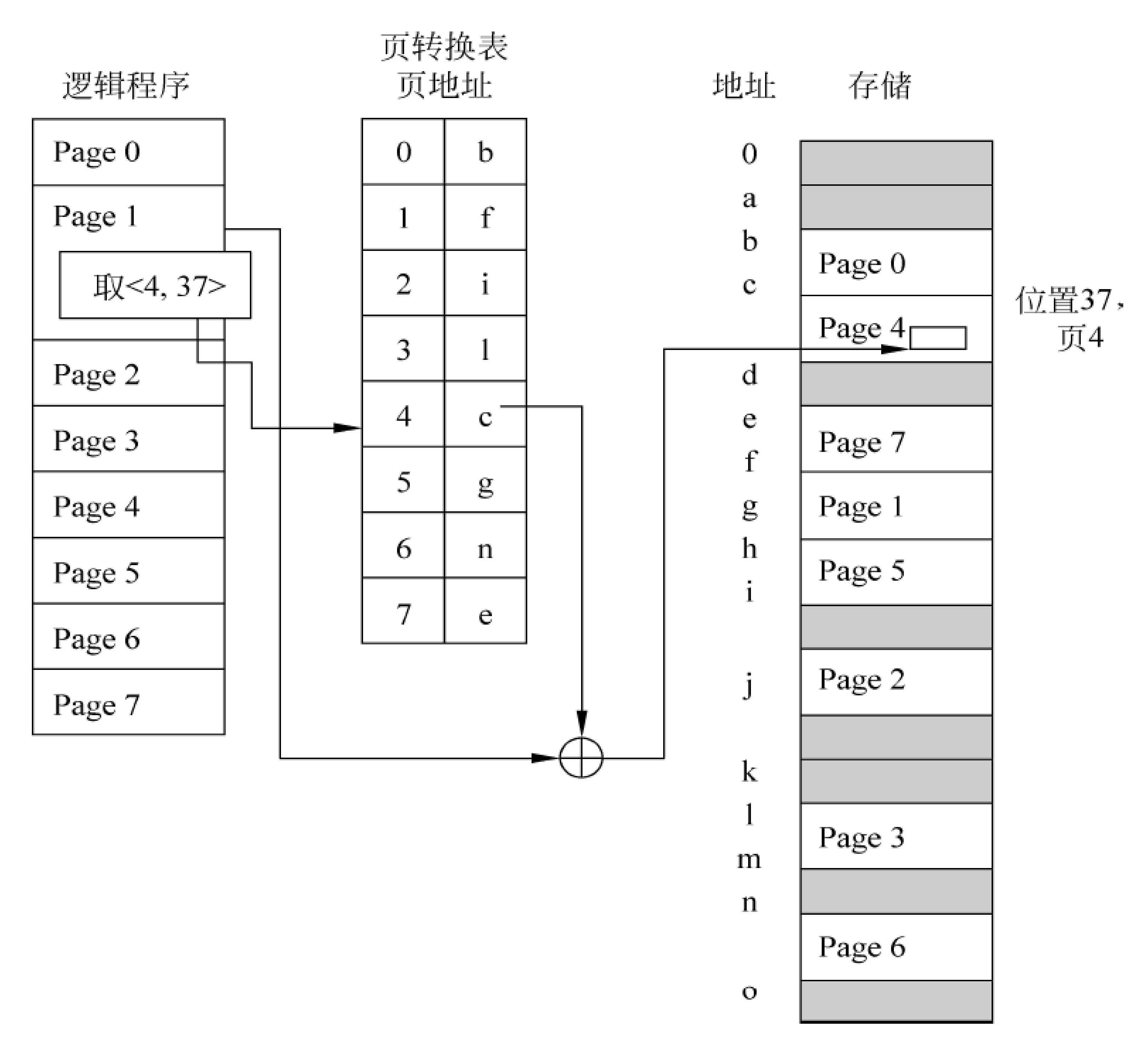


图 3-8　页地址转化

与分段不同，分页方法中所有的页都是相等的固定的大小，所以不存在分段的碎片问题。每个页都可以在内存中找到合适的页，这样就解决了超过页长的地址问题。二元组形式的＜页，偏移量＞地址的设计是为了使偏移量填充地址的一些位。

分页是怎样工作的呢？不妨假设一页的大小为 1024（$=2^{10}$）字节，这里 10b 是分配给每个地址的偏移量，此时程序不能产生超过 1023 字节偏移量。移动到＜x，1023＞下一个位置会引起页部分进位，这样就会移动到下一页。这个转换过程中，分页进程检查以确保＜页，偏移量＞引用没有超过进程定义的页的最大数目。

使用分段法，程序员必须考虑片段。然而，当使用基于分页的操作系统时，程序员会忽

略页界限。此外,分页中每页没有一个逻辑统一体;一页即为程序的下一个 2^n 字节。这样,如果更改程序(比如一条指令的添加),就会将这条指令后面的所有指令后移,从而导致每一页最后的一些字节移动到下一页的开始。这种变换不是程序员需要注意的,因为分页的整个机制和地址转换对程序员是隐藏的。

尽管如此,考虑到保护,这种转换存在一个严重的问题。因为片段是逻辑统一体,因此可以对每个片段实施不同的保护权限,比如只读或只执行,这种转换在地址转换阶段可以有效地进行处理。但是使用分页时每页的条目没有必需的统一体,所以没有方法使得一页上的所有值都有相同的保护等级,比如只读或只执行。

3.2.7 段页式保护

我们已经知道分页提供了执行效率,而分段提供了逻辑保护。因为它们都有各自的缺点和优点,所以可以将它们结合起来使用。

主流系统 IBM 390 系列使用了段页式相结合的保护方式。同样,多路复用信息与计算服务操作系统(multics operating system)在分段上采用了分页法。这两种情况下,程序员可以将一个程序分成逻辑片段,每个片段又被分成固定大小的页。Multics 操作系统中地址使用了 18 位的片段名和 16 位的偏移量,地址被分成 1024 字节的页。图 3-9 所示描述了这个转换过程。这个方法保持了片段的逻辑完整性,并对片段授予不同的保护权限,但是它为每个地址增加了一个另外的转换层。添加的硬件提高了执行效率。

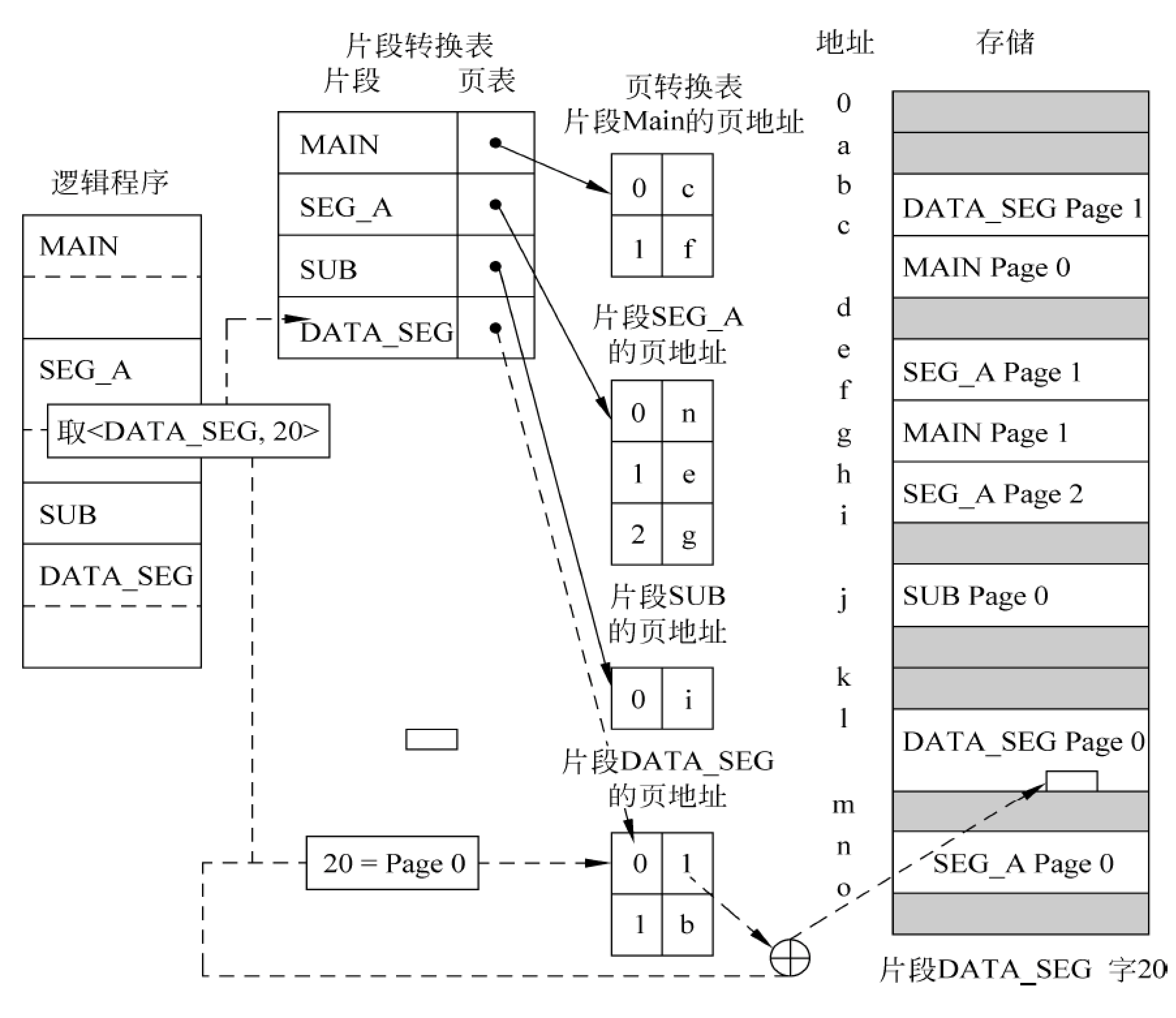


图 3-9 段页式保护

3.3　文件保护机制

3.3.1　保护的基本形式

每个系统都必须提供最小保护来防止用户恶意或无意访问或者修改其他用户的文件。随着用户的增多,这种保护机制也变得越来越复杂。

1. 全无保护

最初的 IBM 操作系统中,文件是默认公开的。任何用户都可以读、修改或删除一个属于任何其他用户的文件。为了取代硬件或软件保护,主要的信任保护与无知联系在了一起。系统设计者假设可以相信用户不会读或修改其他用户的文件,因为用户希望其他用户同样尊重他。这些无知是很有用的,因为用户可以仅通过名字访问文件,假设用户仅知道他们有合法访问权的那些文件的名字。

然而,对于那些敏感的系统文件,系统管理员可以通过口令来保护它们。正常用户也可以使用口令,但是口令看做是保护系统文件的最好的方法。有时,口令控制所有的访问(读、写或删除),使得系统管理员可以完全控制所有的文件。但是有时候口令仅控制写权限和删除权限,因为只有这两个操作会影响其他用户。无论哪种情况,每次访问文件时口令机制都要求系统操作人员的干涉。

然而,这种全-无(all-or-none)保护由于以下几个原因是不能被接受的:

(1) 缺少信任。可靠用户的假设是不合理的。对于用户知道彼此的系统而言,互相尊重是可能的,但是对于大部分系统而言,每个用户都不知道其他用户,他们之间是无法产生信任的。

(2) 太粗糙。即使用户可以区分出一组可靠的用户,也没有一种方法可以使得仅允许这组可靠用户访问。

(3) 共享的提高。这种保护机制更适用于批环境,用户几乎没有可能与其他用户通信,而且用户在不与系统通信时才能进行他们的思考和开发。然而,对于共享的系统,用户与其他用户以及代表不同等级用户的程序通信。

(4) 复杂性。文件保护要求操作人员的参与,这样操作系统的性能就下降了。因此,文件保护这种类型在计算机中并不受欢迎,除了大部分的敏感数据集。

(5) 文件目录。出于审计的目的,也为了帮助用户记住他们负责哪些文件,不同的系统都会产生一个所有文件的目录。这样,用户就不会忘记系统中存在哪些文件。交互用户可能试图浏览不受保护的那些文件。

2. 分组保护

全无保护有许多缺陷,研究人员于是寻求一种改进方法来保护文件,他们致力于将有共同关系的用户分为一个组。比如,在 UNIX 系统中,所有的用户被分成这样几类:用户、可信工作组、合作用户、其他用户。简单地说,可以概括为用户、组和其他用户。Windows NT 系统采用了相似的保护体制,将用户分为管理员、强用户、用户、访客(NT 管理员可以创建其他组)。

所有的授权用户被分成若干组。一个组可能包含工作于同一个项目、部门、班级的成员或单个用户。组关系的基础是存在共享。组成员之间有一些共同的利益，不妨假设他们有需要与其他组成员共享的文件。这种方法中，没有用户属于两个或两个以上的组，否则，同时属于组A和B的成员可能将文件A传递给组B的另一个成员。

当创建一个文件时，用户定义对文件的访问权限，可以是用户自己的，也可以是同一个组的其他成员，或者是所有的其他用户。一般来说，访问权限的选择是一个有限组，比如{更新，只读，读，写，创建，删除}。对于一个特殊的文件，可能授予其他用户只读权，组用户读和更新权，用户所有权限。这个方法将适用于组产生的文件，此时不同的组成员可能修改正在被写入的部分。这个文件本身允许组外的用户浏览但不允许他们进行修改。

组保护法(group protection)一个很突出的优点是它执行的简易性。一个用户通过两个标识符识别：用户ID和组ID。这些标识符存储在每个文件的目录条目中，用户登录时操作系统获得它们。因此，操作系统可以检查请求访问文件的用户的组ID是否与有该文件访问权限的组ID相匹配。

尽管这个保护方案克服了all-or-nothing方案的缺陷，但是它也产生了新的问题。

(1) 组关系。一个用户不能属于两个组。假设Tom与Ann属于一个组，与Bill属于另一个组。如果Tom暗示一个文件对组可读，那么这个组是指哪一个呢？假设Ann的一个文件对组可读，Bill可以访问它吗？解决这些模棱两可问题的最佳方案就是每个用户只属于一个组(这并不是说所有的用户都属于一个组)。

(2) 多重身份。为了克服一个用户一个组的限制，一些用户可能获得多个账户来允许他们成为多个用户。这个漏洞带来了新的问题，因为单个用户在一个时间只能是一个用户。假设Tom获得了两个账户，于是就形成了Tom1与Ann属于一个组，Tom2与Bill属于一个组。Tom1并不与Tom2属于同一个组，所以只有当Tom1的文件、程序或者帮助的访问权限设为world时，它们才对Tom2可用。多重身份导致了账户的增多，冗余的文件，文件的有限保护，而且对用户而言是不便的。

(3) 所有组。为了避免重身份，系统管理员可能决定Tom在活跃状态下可以访问他所有的文件。这个解决方案使得Tom有责任控制他与谁共享什么，他可能与Ann属于Group1，与Bill属于Group2。他创建Group1文件与Ann共享，但是如果下次他登录时在Group2是活跃状态，他仍然看到Group1文件，不过可能没意识到Bill不可访问该文件。

(4) 有限共享。文件只可能在组或world范围内共享。用户希望能够以文件为基础来确认需要共享一个文件的成员，比如与10个用户共享一个文件，而与20个其他用户共享另一个文件。

3.3.2 单一权限

尽管有缺点，这里给出的文件保护方案却是简单的、易懂的。当对单个文件授予权限时，为了简化实施，建议使用其他的易操作的方法来提供更高程度的安全。

1. 持续权限

这个方法通常的实现是使用一个名字，一个令牌，或一个秘密。同样地，计算时根据访问列表授予访问权限，出示一个令牌或票据，或给一个口令。用户访问许可用于所有的访问或仅修改(写权限)。

所有这些方法在撤销时都有一个问题：将某人从一张列表删除很容易，但是找出授权给这个人的列表将其删除就很困难了。重新请求一个令牌或口令则更加困难。

2. 临时获得权限

UINX+操作系统提供了一种很有趣的许可机制，它基于用户-组-其他用户三层结构。UNIX 设计者添加了一个称为 set userid(suid)的许可权限。如果这个保护是为被执行的文件而设计的，那么保护级别就是文件属有者的，而不是执行者的。假设 Tom 拥有一个文件，允许 Ann 使用 suid 执行这个文件。当 Ann 执行这个文件时，她有的是 Tom 的保护权而不是她自己的。

这个特殊的许可很有用，它允许用户建立数据文件，但是只有指定的程序才能访问这些文件。

举一个例子，假设你希望建立一个由计算机管理的约会服务数据库系统，对用户空闲的晚上进行管理。Sue 可能想在周六约会，但是她已经拒绝了 Jeff 的请求，说她有其他计划，Sue 不希望 Jeff 知道她有空。为了使用这个服务，Sue、Jeff 以及其他人必须能够读文件并写入文件(至少不是直接的)，来确定谁有空或公开他们的可用时间。但是 Jeff 能够直接读文件，他会发现 Sue 撒谎了。因此，你的约会服务必须强制 Sue 和 Jeff(以及其他人)仅通过一个访问程序访问这个文件，这个程序可以显示 Jeff 获得的数据。但是如果文件的读写访问仅限于作为管理者的你，Sue 和 Jeff 将永远不可能向其中输入数据。

这个解决方案就是 UNIX SUID 保护。你创建的数据文件，仅给你访问许可。你也可以写程序来访问数据库，然后使用 SUID 保护保存的程序。当 Jeff 执行你的程序时，他临时获得你的访问许可，但是仅在程序执行过程中。Jeff 没有对文件的直接访问权，因为你的程序会进行实际的文件访问。当 Jeff 退出你的程序时，他重新获得他自己的访问权限，失去你的访问权限。这样，你的程序可以访问文件，但是程序必须显示给 Jeff，只有这个数据允许 Jeff 查看。

这个机制使用很方便，普通用户只能用规定的方法实现。比如，仅系统能够使用用户口令修改文件，但是每个用户都可以随时改变他们自己的口令。使用 SUID 特性，口令修改程序权归系统所有，因此系统对系统口令表有完全的权限。改变口令的程序也由一个 SUID 保护，这样当一个正常用户执行程序时，程序能够用受限的方式以用户的立场修改口令文件。

3.3.3 每个对象与每个用户保护

这些保护方案的主要局限是创建与用户有联系且有意义的组的能力，这些用户有着访问一个或多个数据集的相似的访问权限。访问控制列表或者访问控制矩阵描述了早期提供的灵活的保护。他们的缺点是对于那些想获得许多用户以及许多不同的数据集的访问权限的用户而言的；这样的用户必须对允许每个用户访问的每个数据集进行具体说明。作为一个新添加的用户，用户特殊的访问权限必须对所有合适的用户进行说明。

3.4 用户认证

一个操作系统的大部分保护是基于系统的用户而言的。在实际生活中，人们常常询问他们不知道的人们的身份：银行雇员可能在兑现一张支票前要求出示驾驶证，图书馆雇员

可能在还书前要求身份证明，移民局可能会要求护照来作为身份的证明。这种面对面的身份证明比远程的身份证明容易多了。比如，一些大学不会通过电话通知成绩，因为办公人员不知道是否是那个学生在接听。然而，一个能够识别学生声音的教授可以告诉学生他的成绩。认证的方法很多，文件、声音识别、指纹识别等都可以用来进行身份认证。

计算时，选择受限更多，安全性更低。任何人都可以试图登录一个计算系统。与能够识别学生声音的教授不同，计算机不能区分不同用户的电子信号的不同。这样，大部分的计算认证系统必须基于一些仅被用户和计算系统共享的知识。

认证机制可以使用下面三种方法之一来确认用户的身份：①用户知道的事情，口令、PIN 数字、秘密的握手以及母亲的名字都是用户可能知道的事情；②用户拥有的物品，身份标志、公钥、驾驶证或者制服都是一些常见的可以区分用户身份的物品；③用户的所属，这些证明者称为生物特征，以用户的物理特征为基础，比如指纹、用户声音的形式、人脸（图像）。这些认证方法都是旧的（人们通过朋友的脸在人群中辨认他们，通过他们的声音在打电话时区分他们），但是它们刚刚开始用于用户与计算机之间的认证。

将这三种方法进行组合可以形成更强的认证，比如，银行卡和 PIN 就将用户拥有的物品与用户知道的事情进行了组合。

口令是相互协商所得的码字，假设仅被用户和系统知道。在一些情况下，用户选择口令；在另一些情况下，系统分配口令。口令的长度和格式也因系统的不同而不同。

为了帮助用户选择好的口令，一些系统提供无意义但可以读的口令。比如，VAX VMS 系统随机产生 5 个口令，用户从中选择一个。它们可读出来，所以用户能够重复并记住它们。然而，用户可能记错一个口令，改变内部的音节或无意义的字符串中的字母（bliptab 很容易被记成 blaptib 或 blaptip）。

有时系统强迫用户定期修改口令，每次要求修改口令时，拥有最喜欢的口令的用户会在两个口令间作出选择。为了避免口令重用，Microsoft Windows 2000 系统拒绝接受任何最近 k 次使用过的口令。这种系统的用户每月需要经历 24 次口令更换才能返回到他最喜欢的口令。

一次口令（one-time password）是指使用一次就更换的口令。系统分配给用户的不是一个静态的短语而是一个静态的数学函数。系统提供参数给函数，用户进行计算并返回函数值。这样的系统也称为挑战响应系统（challengeresponse system），因为系统给用户一个挑战，并通过用户的响应判断用户的真实性。

一次口令在认证中有很重要的地位，因为截获的口令是无用的，它不能被重用。然而，它们的有用性受到了算法复杂性的限制，人们希望算法容易被记住。口令生成器设备可以执行很复杂的函数，并且有一些价格合理的模型，它们能够有效地抵抗在网络中以明文形式传输口令的危险。

用户有时会误输他们的口令。用户在接收到不正确登录的消息后会仔细地重新登录，并获得系统的访问权。即使是很糟糕的打字员在尝试多次后也能够成功登录。

一些认证程序很慢。如果登录过程需要 5 秒或 10 秒，合法用户不会抱怨。对于进行穷举搜索或字典搜索的密码分析员而言，每次 5 秒或 10 秒会使得这种攻击不可行。

那些总是登录失败的用户可能不是授权用户。当连续几次登录失败后，系统常常中断连接，强迫用户重新与系统建立连接。

在更安全的装置中，阻止攻击者比容忍用户错误更为重要。比如，一些系统管理员假设所有的合法用户在三次以内能够正确输入他们的口令。连续三次口令失败后，用户不能使用那个账户，只有安全管理员才能使这个账户重新使用。这种识别账户的方法可能成为攻击者的目标。

口令认证假设知道口令的人是口令的所有者。正如平常所看到的，口令可以被猜测或者推断。一些人在询问时给出他们的口令。其他的口令只有在看到用户输入时才能获得。口令可以被认为是证据的第一部分，但是怀疑者会想要更多的证据。

有许多方法可以用来提供二级保护，包括另一轮口令或挑战响应交换。

1. 挑战响应系统

登录常常是时间不变的。除了口令更改时，每次登录看起来都和以前一样。更复杂的登录要求用户 ID 和口令，以及一个挑战响应交换。在这样的交换中，系统给用户一个回复，每次用户登录时这个回复都不同。比如，系统可能显示四位数字，而用户必须准确输入一个函数，比如求解这些数字的乘积或和的函数。每个用户被分配一个不同的挑战函数来计算，因为有许多可能的挑战函数，获得用户 ID 和口令的攻击者不能推算出正确的函数。

与计算器相似的物理设备可以用来执行更复杂的响应函数。用户输入挑战数字，设备计算并显示响应给用户登录。

2. 冒充登录

这个系统中，证据是单向的。系统要求用户身份证明，但是假设用户信任系统。不过，程序员很容易写一个程序来显示用户 ID 和口令的提示，捕获输入的对并存入文件中，然后显示“系统错误；中断连接”，最后退出。这种攻击是特洛伊木马的一种类型。攻击者建立后便不再管理终端，而是等待无辜的受害者登录。此时的受害者根本想不到已经出现了一个安全缺陷。

为了抵抗这种类型的攻击，用户必须保证每次使用系统时系统的访问路径都重新初始化。在一些系统中，再次打开或关闭终端或者按 Break 键产生一个信号给计算机使其暂停终端进程的运行。然而，不是每台计算机都能辨认出 power-off 或者 Break 是对当前进程的中断。计算系统常常通过网络访问，所以复杂的初始化是不可能的。

用户可能怀疑计算系统，就像系统怀疑用户一样。在确信计算系统是合法的之前，用户不会输入秘密数据（比如口令）。当然，只有通过认证过程，计算机才会承认用户。计算系统可以显示一些仅被用户和系统知道的信息。比如，系统可能读用户的姓名，并回复“你最后一次登录是 4 月 10 日 9：47”。用户在输入秘密的口令之前能够验证日期和时间是否正确。如果要求更高的安全，系统可能发送加密的时间戳。用户对其解密，并发现时间是当前时间。为了使系统确信恶意的攻击者没有从以前的登录中截获口令，用户以加密的时间戳和口令进行回复。

认证讨论中时常被忽略的可靠性也是一个双边问题：系统需要确认用户是真实的，但是用户也同样需要确认系统。第二个问题导致了计算机欺骗的另一个类型——网络钓鱼（phishing），用户提交敏感信息给一个冒充可信任系统的恶意系统。网络钓鱼攻击的目标是银行和其他金融机构，因为欺骗者可以使用他们从客户获得的敏感数据从真正的机构取

走用户的钱。

认证在操作系统中有着很重要的作用，因为准确的用户鉴别是个人访问权限的关键。大部分操作系统和计算系统管理员采用的是合理但严格的安全措施，从而在非法用户访问系统之前将其排除，但有时会将一些不合适的机制强制用作认证设备。

3.5 系统行为审计

审计是对计算机系统和网络进行安全评估和保证的工具。入侵检测时用户正在进行的活动的某个记录必须保存下来，通常有两个方案。

(1) 本地审计记录：事实上，所有的多用户操作系统都包括收集用户活动信息的审计软件。使用这个信息的优点是不需要另外的收集软件，缺点是本地审计记录可能不包含需要的信息或不以方便的形式包含它。

(2) 检测明确的审计记录：收集设备能够产生仅包含入侵检测系统要求的信息的审计记录。这种方法的优点是供应商独立，可以用于多个不同的系统，缺点是产生额外的开销，事实上，两个审计包运行于一台机器上。

一个检测明确的审计记录应包含以下几个域。

(1) 主体：行为的发起者。主体一般是一个终端用户，但也可能是代表用户或群用户利益的进程。所有的活动由主体的指令产生。主体可能被分成不同的访问等级，这些等级可能重叠。

(2) 行为：主体对客体的操作，如登录、读、执行 I/O、执行。

(3) 客体：行为的接收者，包括文件、程序、消息、记录、终端、打印机以及用户或程序创建的结构。当主体是行为的接受者时（比如电子邮件），主体被看做是客体。客体可能被分成各种类型。客体粒度可能因客体的类别和环境的不同而不同，比如，数据库行为可能作为整个数据库被审计，也可能作为记录被审计。

(4) 异常条件：当返回时会出现异常条件，可用 which、if any 表示。

(5) 资源使用：定量元素表，其中每个元素表示某个资源的使用量，如打印队列的数目、读或写的记录的数目、使用的 I/O 单元、处理时间等。

(6) 时间戳：当行为发生时，用于标识的唯一的时间-日期标志。

一个完全的审计应该包括以下几个步骤：①检查系统描述的标准的每个方面；②检查所有识别的威胁；③选择一个审计频率，可以是每天、每星期或每月；④检查实际操作，保证与写的准则一致。

大部分用户操作由许多基本行为组成，比如，文件复制涉及用户指令的执行，包括访问权限验证和设置复制、从文件中读以及写入另一个文件。下面给出了一条指令，它表示 Smith 从当前目录将文件 GAME 复制到＜Library＞目录：

COPY GAME. EXE TO＜Library＞GAME. EXE

可能会产生这样的审计记录：

Smith	execute	＜Library＞COPY. EXE	0	CPU＝00002	11058721678
Smith	read	＜Smith＞COPY. EXE	0	RECORDS＝0	11058721679
Smith	execute	＜Library＞COPY. EXE	write-viol	RECORDS＝0	11058721680

当 Smith 没有写入＜Library＞的许可时，复制被取消。

将用户的操作分解成基本行为有三个优点：

（1）因为客体是系统中被保护的条目，使用基本行为能够对影响客体的所有行为进行审计。这样，系统不但可以检测出试图破坏的访问控制（通过观察大量返回的异常情形的反常），而且可以检测出破坏成功的访问控制（通过观察一组客体访问主体的反常）。

（2）单个客体，单个行为审计记录简化了模型和实现。

（3）因为简单性以及检测明确的审计记录的统一结构，信息的获得可能会相对容易一些，至少一部分可以直接通过存在的本地审计记录和检查明确的审计记录之间的映射得到。

3.6　UNIX 安全

UNIX 是一种多用户、多任务的操作系统，这类操作系统的一种基本功能就是防止使用一台计算机的不同用户之间互相干扰。

UNIX 系统具有两个质形态：核心态和用户态。运行内核中程序的进程处于核心态，运行核外程序的进程处于用户态。系统保证用户态下的进程只能存取它自己的指令和数据，而不能存取内核和其他进程的指令和数据，并且保证特权指令只能在核心态执行，像中断、异常等在用户态下不能使用。用户程序可以通过系统调用进入核心，运行系统调用再返回用户态。系统调用是用户在编写程序时可以使用的界面，是用户程序进入 UNIX 内核的唯一入口。因此，用户对系统资源中信息的存取都要通过系统调用才能完成。一旦用户程序通过系统调用进入内核，便完全与用户隔离，从而使内核中的程序对用户的存取请求进行响应，而不受用户干扰的访问控制。

3.6.1　UNIX 保护体制

UNIX 保护体制有保护态和一组能够使进程对状态进行修改的操作，所以 UNIX 是 DAC 系统。UNIX 保护态将进程身份（主体）与它们对文件的访问权限（客体）联系起来。每个 UNIX 进程身份包括一个用户 id（UID），一个组 id（GID）以及一个附加组（supplementary groups）。

所有的 UNIX 资源都表示为文件。保护态使用这些操作的标准含义，表明主体可能对文件实施读、写和执行操作。当目录不是文件时，它们表示的是 UNIX 保护态中的文件，尽管操作有不同的语义（如执行指搜索目录）。

文件也将所有者 UID 与所有者 GID 联系起来，并传递给拥有这些身份的进程。一个

拥有所有者 UID 的进程能够修改这个文件保护态的所有方面。拥有所有者 UID 或所有者 GID 的进程可能拥有另外的对文件的访问权限。

客体和操作的有限性使得 UNIX 设计者使用压缩的访问控制列表(称为 UNIX 模式位)来说明主体对文件的访问权限。模式位将主体的权限定义为三类：文件所有者 UID，文件组 GID 以及所有的主体。使用模式位授权的执行过程如下：首先，UNIX 授权机制检验进程身份的 UID 是否与文件的所有者 UID 一致，如果是，那么使用所有者的模式位授权，如果进程身份的 GID 或者附加组与文件的组 GID 一致，那么使用组权限的模式位，否则，权限被分配给所有其他使用者。

举一个例子。UNIX 模式位的形式为{所有者位，组位，其他位}，该元组中的每个元素包含一个读位，一个写位以及一个执行位。模式位：

rwxr--r--

表示拥有相同的所有者 UID 的进程能够读、写或者执行文件，拥有与文件的组一致的 GID 或者附加组的进程能够读文件。

假设一组文件拥有下列的所有者、组以及描述如下的模式位：

Name	Owner	Group	Mode Bits
foo	Alice	faculty	rexr--r--
bar	Bob	students	rw-rw-r--
baz	Charlie	faculty	rwxrwxrwx

这样，使用组 faculty 运行的进程 Alice 能够读、写或者执行 foo 和 baz，但是仅能够读 bar。对于 bar，Alice 与 UID(Bob)不匹配，也与组(students)无联系。这个进程由合适的所有者来获得 foo 的所有特权以及合适的组来获得 baz 的特权。

如上所述，UNIX 保护体制是一个自主访问控制体制，这意味着文件的模式位、所有者 UID、或者组 GID 都可能被文件所有者运行的任何 UNIX 进程改变(比如与文件所有者拥有相同的 UID)。如果相信所有的用户进程都是为了用户的最大利益，那么用户的安全目标能够被加强。然而，这并不是合理的假设。现在，用户运行各种各样的进程，其中可能有些是黑客提供的，还有一些可能很容易被黑客攻破，所以用户无法保证这些进程与用户的安全目标一致。因此，安全操作系统不能使用自主访问控制来加强用户的安全目标。

安全自主访问控制允许用户改变他们文件的所有者 UID 和组 GID，而且模式位和文件标识也是自主的。一个安全保护体制要求强制标识态，所以这是 UNIX 系统无法满足安全操作系统的另一个原因。

UNIX 进程通过来自一组由可信管理员定义的标识的信任服务进行标识，子进程继承父进程的身份。这种使用身份的标识进程的强制法能够满足安全保护体制的要求，不过这个方法的灵活性差。

最后，UNIX 模式位还包括保护域转换的具体说明，称为 setuid 位。当这个位用于文件时，任何执行该文件的进程自动实施对文件的所有者 UID 和组 GID 的保护域转换。比如，如果 root 进程对一个它拥有的文件设置 setuid 位，那么任何执行这个文件的进程都会以 root UID 执行。因为 setuid 位是一个模式位，它能够被文件的所有者设置，所以它也可以用自主方式管理。一个安全的保护态要求强制转换态描述所有的保护域转换，所以自主 setuid 位的使用无法满足要求。

3.6.2　UNIX 授权

UNIX 授权机制控制每个进程对文件的访问，并实施保护域转换使得进程能够改变它的身份。授权机制在内核运行，但是它依赖于系统和用户进程来决定它的授权请求和保护态。

当文件打开时，UNIX 授权会发生，每次访问文件时验证允许对文件的操作。请求进程提供文件名和对文件的操作，如果授权，UNIX 创建一个文件描述符来表示授予进程的权限(实施将来对文件的操作)。文件描述符存储在内核中，仅返回给进程一个索引。这样，文件描述符是能力表的形式。当用户请求对他们打开的文件进行操作时，用户进程表示它们对内核的文件描述符索引。

UNIX 授权通过仲裁打开文件的读、写和执行权限控制传统的文件操作。然而，这些权限的使用并不总是达到期望的效果：①这些权限和它们的语义并不总能满足控制；②一些客体不表示文件，所以它们不能仲裁。如果用户拥有对文件的读权，这足够他对文件实施除了读以外的各种操作。比如，拥有文件描述符，用户进程使用系统请求 ioctl 或者 fcntl 以及读和修改元数据，能够实施任何对文件的特别指令。此外，UNIX 不仲裁所有的安全敏感客体，比如网络通信。主机防火墙提供网络通信的某种控制，但是它们不通过进程身份限制网络通信。

UNIX 授权机制依赖于用户级别的授权服务(如登录和 sshd)来确认进程身份(UID，GID 和附加组)。当用户登录系统时，他的进程被分配给他的登录身份。除非有一个域转换，否则所有这次登录过程中创建的进程继承这个身份。这样的用户级别的服务也需要 root 特权来改变进程的身份，所以它们使用特别的 UID 运行。不过，一些 UNIX 服务需要作为 root 运行来获得执行任务所需要的特权、访问系统文件和目录、改变文件许可等。这些服务中的某些对于改正 UNIX 授权的操作很重要，比如 sshd 和 passwd，但是其他的就没那么重要了，比如 inetd 和 ftp。尽管如此，UNIX 系统的可信计算基必须包含所有的 root 进程，否则会给安全关键服务和内核本身带来风险。

UNIX 保护域转换是通过 setuid 机制实现的。有两种方式使用 setuid：①root 进程发起 setuid 系统请求来改变进程的 UID；②文件使用它的 setuid 模式位，这样任何时候执行文件时，文件的身份都设置为文件所有者。第一种情况下，特权进程(如登录和 sshd)能够改变进程的身份。比如，当用户登录时，登录程序必须改变用户第一个进程的进程身份以及他的 shell，以保证正确的访问控制。第二种情况下，文件的 setuid 位一般用来许可低特权条目执行高特权程序，此时几乎都是作为 root 使用的。比如，当用户希望改变他的口令时，他使用 passed 程序。因为 passwd 程序修改口令文件，它必须具有特权，所有使用用户的身份运行的程序不能改变口令文件。为 root 所有的，passed 执行的文件设置 setuid 位，当任何用户执行 passwd 时，由此产生的进程身份转换为 root。当身份转换不影响用户的其他进程时，passed 程序的写方必须禁止程序允许用户控制 passed 怎样使用它附加的特权。

UNIX 也有一对机制可以使得用户使用减少的许可运行进程。但是，这些机制很难正确使用，也仅对 root 进程可用，而且仅能进行适当的限制。首先，UNIX 系统有一个特殊的主体 nobody，它不拥有任何文件，也不属于任何组。因此，进程的许可仅限于 nobody 运行，因为它永远没有所有者和组特权。不幸的是，nobody 和所有的主体一样，有其他的特权。

而且,因为仅 root 能够设置 setuid,只有超级用户进程能够改变 nobody 的进程身份。其次,UNIX 可以使用 chroot 限制进程对文件系统子树的访问。这样,进程仅能访问它子树的文件。不幸的是,必须设置 chroot 环境来防止进程逃避受限的域。比如,如果攻击者能够创建子树的/etc/passwd 和/etc/shadow 文件,那么他能够添加一个 root 条目,作为 root 登录,避免 chroot 环境(使用 root 访问内核)。而且,一个 chroot 环境仅能由 root 进程设置,所有一般的系统用户都不可用。实际上,这些方法都不是限制进程权限的有效方法。

3.6.3 UNIX 安全分析

如果 UNIX 是一个安全操作系统,它必须满足安全操作系统的要求,但是,下面这些要求 UNIX 不满足。

(1) 完全仲裁:访问监控保证(the reference monitor guarantees)如何保证所有的安全敏感操作都被正确仲裁?

UNIX 访问监控保证由钩组成,它们检查文件访问权或者系统请求的索引节点许可。UNIX 访问监控保证授权对客体(内核在它的操作中使用)的访问权。

问题是 UNIX 操作(读、写和执行)有限集无法表达所有的访问控制信息,UNIX 不需要写权限就允许修改文件(如 fcntl)。

(2) 完全仲裁:访问监控保证对所有的安全敏感操作进行仲裁吗?

UNIX 授权不提供对所有系统资源的完全的仲裁。对于某些客体,比如网络通信,UNIX 本身完全不提供授权。

(3) 完全仲裁:我们怎样验证访问监控保证提供了完全仲裁?

因为 UNIX 访问监控保证被安置于被实施的安全敏感操作,很难知道是否所有的操作都被标识以及所有的路径都被仲裁。没有具体的方法被用来验证完全仲裁。

(4) 预防篡改:修改时系统怎样保护访问监控器,包括它的保护体系?

访问监控器和保护体制都存储在内核中,但是这不能保证防止篡改。首先,保护体制是自主的,所以它可能被任何运行的程序篡改。非信任的用户进程能够随意修改它们的用户数据许可,所以加强用户数据的安全目标是不可能的。

其次,UNIX 内核不受非信任用户进程的保护。虽然 UNIX 使用保护环进行隔离,但是 UNIX 内核常常提供程序验证系统请求的参数,这样程序可能被放错位置。

最后,用户级别的进程有各种各样的接口访问和修改内核本身而不需要系统请求,包括通过 netlink 套接口安装内核模块到具体的系统文件(如/proc 或者 sysfs)的接口,以及直接访问核心内存(如通过设备文件/dev/kmem)。保证这些接口仅能被可信代码访问就变得不实际了。

(5) 预防篡改:系统的保护体系保护可信计算基程序吗?

UNIX 可信计算基(TCB)由所有的 root 进程组成,包括所有的以 root 用户登录的用户运行的进程。因为这些进程能够运行任何程序,保证 TCB 不被篡改是不可能的。即使忽略 root 用户,TCB 代码的数量也太大了,要求一个预防篡改的可信计算基需要面对更多的威胁。比如,一些 root 进程有开放的网络端口,它们可能被用来作为破坏进程的通道。如果这些进程中的任何一个被攻破了,UNIX 系统也就被攻破了,因为 root 进程中没有任何有效的保护。

此外,任何 root 进程都能修改保护体制的任何方面。UNIX 的 root 进程可能无法完全

信任或保护,所有保护体制的未授权的修改是可能的。因此,在 UNIX 系统中不能依赖于预防篡改保护体制。

(6) 可证明:系统的可信计算基的正确性的根据是什么?

UNIX 系统的任何正确性的基础都是不规范的。TCB 的无限大小阻止了任何有效的形式化证明。不仅如此,内核的大小和可扩展的属性(如新的设备驱动和其他内核模块)也使得无法验证它的正确性。

(7) 可证明:保护体系加强了系统的安全目标吗?

安全目标的可验证性的加强是不可能的,因为缺少完全仲裁和预防篡改。因为不能用一种很好的策略来防止未授权数据泄漏和修改,所以我们不能加强安全内核的机密性和完整性。因为不能证明 TCB 能够防止被攻击,所以不能证明系统是一直在加强我们想要达到的安全目标,即使它们能够很好地表示。

3.6.4　UNIX 漏洞

安全操作系统必须保护它的可信计算基被破坏,这样也是为了保证安全检测的实施。下面我们将列出 UNIX 系统的一些导致 UNIX 的可信计算基被破坏的安全漏洞。

1. 网络守护进程

UNIX 有一些 root(如 TCB)进程能够保持网络端口对所有的远程控制方开放(如 sshd,ftpd,sendmail 等),称为网络守护进程(network-facing daemons)。为了保护系统的可信计算基的完整性,确保实现访问监控,这样的进程必须免受攻击。然而,这些进程存在一些漏洞,容易受到缓冲区溢出的攻击,它使得远程攻击者能够攻击系统的 TCB。这些守护进程已经被重新设计以避免这样的攻击(如 Postfix 取代了 sendmail 和特权分离的 SSH),但是修改后的保护进程不提供完全的完整性保护。这样,UNIX 中网络守护进程的完整性保护就变得不全面且很特别。

此外,一些网络守护进程,比如远程登录守护进程(如 telnet,rlogin 等)ftpd 和 NFS,它们在网络中设置了可取消的信任。远程登录守护进程和 ftpd 都因为以明文形式传送口令而臭名昭著。幸运的是,这样的守护进程已经过时了,它们已被新的安全版本取代(如 vxftpd 取代 ftpd)。同样,NFS 因为把任何远程文件系统的请求的响应都作为合法服务器的响应来接收而名声很坏。网络守护进程还必须保护它们秘密的完整性和认证远程数据资源,这些数据的完整性对进程很重要。

2. 隐藏软件

现代 UNIX 系统支持扩展,通过可被直接装入内核中的内核模块实现。然而,恶意模块可能使得攻击者能够使用完全的系统特权执行内核中的代码。隐藏软件(rootkit)是各种各样的恶意软件包,它们能够利用内核模块的载入或者其他内核接口使用 root 进程。这样的 rootkit 能够帮助攻击者进行攻击,且提供了入侵的措施。尽管采用了很多方法来检测内核中的恶意软件,但是这样的 rootkit 一般很难检测到。

3. 环境变量

UNIX 系统支持环境变量(environment variables),即使得进程能够通过应用程序传递变量的系统变量。LIBPATH 就是这样的一个变量,它的值决定了动态库的搜索顺序。一

个常见的漏洞是攻击者能够改变 LIBPATH，从而将攻击者提供的文件载入作为动态库。因为当子进程被创建时，环境变量被继承，所以非信任进程能够在非授权环境中请求 TCB 程序。如果 TCB 进程依赖于动态库，自己不设置 LIBPATH，它可能很容易运行恶意代码。因为很多 TCB 程序能够通过 setuid 请求，这是一个普遍的问题。

此外，TCB 程序可能受到任何非信任进程提供的输入值的攻击，比如恶意输入的参数和多个程序允许请求者定义进程的配置文件。配置文件一般描述所有的其他部分，程序应该寻找输入来描述它怎样运行，有时包括它需要使用的库的位置和提高网络信息的主机的位置。如果攻击者能够控制程序配置文件的选择，他常常有很多方法攻击运行的进程。任何 TCB 程序必须保证它们的完整性而不管它们的请求方式。

4. 共享资源

如果 TCB 进程使用非信任进程共享资源(shared resources)，那么它们可能很容易受到攻击。常见的问题是/tmp 库的共享。因为任何进程都能够在这个目录下创建文件，所以非信任进程能够在这个目录下创建文件并授权其他进程，甚至一个 TCB 进程也能访问这样的文件。如果非信任进程能够猜测 TCB 进程的/tmp 文件的名字，它便能够创建这个文件，授予这个 TCB 进程的访问权限，然后使自己访问 TCB 文件。创建时检查是否存在这样的文件(如使用 O_CREAT 标志)，TCB 进程能够防止这个问题。然而，程序员常常忘记这些防护。当使用任何非信任进程共享的客体时，TCB 进程必须提高警惕。

5. 时差攻击

UNIX 易受到各种攻击，此时非信任进程可能在授权操作与实施操作之间改变系统状态。如果这样的改变使得非授权进程访问未获得授权的文件，就出现了一个漏洞。这个攻击是由 Dilger 和 Bishop 首先提出的，他们称这种攻击为时差攻击(time-of-check-to-time-of-use attacks)。举一个例子，root 进程使用系统请求 access 来决定进程正在运行的用户是否可以访问文件/tmp/X。然而，在 access 系统请求授予文件访问权后，在文件打开前，用户可能改变文件名与实际访问的文件客体(如 inode)的关联，他可能将文件/tmp/X 改变为链接到目标文件/etc/shadow。不过，UNIX 文件系统允许时差攻击，因为匹配文件名和实际的文件客体(inodes)可能被非信任进程实施。

因为自主保护体制，系统 TCB 的大小，以及这些攻击，将 UNIX 系统转变为安全操作系统是一个很重要的挑战。确保 TCB 进程保护它们自身，从而防止访问监测器被篡改是很复杂的，因为非信任进程能够控制 TCB 进程怎样请求并以多种方式提供输入：网络、环境和参数。此外，非信任进程可能使用系统接口操作任何共享资源，甚至可能改变客体名和实际的客体之间的关联。虽然已经做了很多工作来克服这些缺陷，但是 UNIX 系统的复杂性以及它们的可信计算基也使得很难满足访问监测概念中防篡改和可证明的要求。

3.7 Windows 安全

这一节将详细描述 Windows 访问控制系统。与 UNIX 一样，Windows 操作系统也不满足安全操作系统的要求。

3.7.1　Windows 保护体制

Windows 2000 保护体制，与 UNIX 保护体制一样，提供无条件访问控制模型来管理保护状态、客体分类，以及主要的转化。这两种保护系统的不同在于灵活性（Windows 体制可扩展）与表达能力（Windows 体制能够描述更多的策略）。然而，在将 Windows 保护体制与安全保护体制的定义进行比较时，可以发现虽然 Windows 体制提高了灵活性和表达能力，但却使得它的系统安全更难实现。

具体地说，Windows 保护体制与 UNIX 的不同主要是客体、操作以及它分配给主体的额外的灵活性的不同。当 Windows 2000 访问控制模型被发展时，各种不同的安全也在被发展，它们提供管理员扩展的策略语言，这种语言允许灵活的策略说明，如 Java 2 模型。这些模型能够表示所有的保护态，解决了 UNIX 模型的一些缺陷，但它们并不保证系统的安全。

Windows 中的主体与 UNIX 的主体相似。Windows 分配给每个进程一个令牌来描述进程的身份。进程的身份有用户安全鉴别者（主要的 SID，类似于 UNIX UID）、组 SID（不是单个 UNIX GID 和很多附加的组）、冒名 SID（能够以其他身份活动），以及特权（特设的特权只与这个令牌有关）。Windows 的身份仍然与单个用户身份相关，但是用户的进程令牌可能包含权限的所有组合。

与 UNIX 不同，Windows 客体能够有除了文件的许多不同的数据类型。事实上，应用可能定义新的数据类型，并且将它们添加进活动目录中，系统知道所有客体的层次名字空间。从访问控制角度看，客体类型是由它们的操作定义的。Windows 模型也支持客体类型进行操作。Windows 为每个客体类型定义多达 30 个操作，包括一些针对数据类型的具体操作。与 UNIX 保护态的读、写和执行操作相比，这显而易见。即使是文件客体，Windows 保护体制也定义了许多操作，比如访问文件属性的操作和使文件同步的操作。另外，应用可能添加新的客体类型，定义它们自己的操作。

Windows 与 UNIX 保护态之间其他的不同主要是 Windows 支持自主访问控制列表（Arbitrary Access Control Lists，ACLs），而不是 UNIX 的有限的模式位法。一个 Windows 的 ACL 存储一组访问控制项（Access Control Entries，ACEs），描述一个 SID（用户、组或假名）能够对客体执行哪些操作。ACE 的操作基于目标客体的客体类型，在 Windows 中，ACE 可能授权也可能拒绝否认一个操作。与 UNIX 不同，Windows 使用的是消极的访问权限，因此，它们的认证机制有一些区别。

3.7.2　Windows 授权

Windows 授权请求是由一个称为安全访问监控器（Security Reference Monitor，SRM）的部件处理的。SRM 是一个核心部件，带有进程令牌，一个客体 SID，一组操作，它返回认证查询的真假结果。SRM 使用对象 SID 从它决定的查询结果中检索它的 ACL。

因为消极的许可，SRM 处理认证查询的方式比 UNIX 中更为复杂。主要的不同在于 ACL 中的 ACE 是按照一定顺序的，ACE 按照那个顺序被检查。SRM 搜索 ACE 直到它发现一组允许这个操作的 ACE 或一个拒绝这个操作的 ACE。如果 ACE 授权必需的操作，那么这个请求是被授权的。然而，如果遇到一个拒绝的 ACE（包括某个请求操作），那么所有

的请求都被拒绝。

Windows 的仲裁是由一组对象管理器决定的。与 UNIX 中请求访问同类对象的体系不同，Windows 中的每个对象类型都有一个执行该类型的函数的对象管理器。从模块化角度来看，这是很有利的，但是事实是对象管理器可能扩展系统，这就会给仲裁带来一些挑战。我们需要知道每个新的对象管理器仲裁所有的操作，并决定这些操作的权限，Windows 中是没有能够保证这一点的进程。

Windows 可信计算基由所有的系统服务和作为可信用户运行的程序（如管理员）组成。Windows 提供一组类似 uid 的机制请求事先定义了权限的 Windows 服务，至少足够支持所有的客户。否则，这些服务的脆弱性就会导致系统崩溃。此外，软件的易安装性和无条件的 Windows 访问控制模型的复杂性常常导致用户以管理员的身份操作。这种情况下，任何用户程序都能够控制系统。这是 Windows 系统常见的问题。随着 Windows Vista 的发布，Windows 模型被扩展，能够阻止程序从 Internet 下载、自动写入 Windows 应用和 Windows 系统，而不管用户的进程身份。虽然这提供完整性保护，它不能完全地保护系统的完整性。它阻止低完整性的进程写入高完整性的文件，但是不阻止请求、恶意请求或者使用低完整性的文件欺骗高完整性的代码。

Windows 也提供限制灵活使用进程的方法，称为受限上下文（restricted contexts）。通过为进程定义一个受限的上下文，执行一个操作的很重要的权限是必须对使用它的令牌的进程以及受限上下文可用。即受限上下文中运行进程的权限是受限上下文和进程一般权限的交集。因为一个受限上下文可能被分配一组任意的权限，这个机制比无人运行的 UNIX 选择灵活多了。而且，因为受限上下文设在访问控制系统中，与 UNIX 的 chroot 相比，易出错的概率降低了。然而，管理员很难正确定义受限的上下文，所以受限上下文不经常被用户群体使用。

3.7.3 Windows 安全分析

尽管 Windows 访问控制模型提供额外的表达力，但是它也不能满足任何的访问监控保证（the reference monitor guarantees）。虽然 Windows 能够表达任何权限间的关联，但是这对于管理员来说更加困难。几乎没有用户使用 Windows 许可模型，但是大多数用户至少知道怎样使用 UNIX 模型（尽管不都是正确的）。Windows 并不比一般的 UNIX 更安全，它们都是不安全的。

(1) 完全仲裁：访问监控保证如何保证所有的安全敏感操作被正确仲裁呢？

在 Windows 中，仲裁是由对象管理器员提供的，没有资源码，很难知道在哪里执行仲裁，但是假设同 UNIX 相似，对象管理器授权给使用于安全敏感操作的实际对象。

(2) 完全仲裁：访问监控保证对所有的安全敏感操作进行仲裁吗？

对象管理器提供完全仲裁的机会，但是不提供没有保障的仲裁。此外，一组管理器可能被扩展，导致潜在的不安全管理器的增加。如果没有一个规范的方法来定义每个管理器做什么、怎样才能不受攻击，提供完全仲裁的保障是不可能的。

(3) 完全仲裁：我们怎样验证访问监控保证提供了完全仲裁？

对 UNIX 而言，没有具体的方法被用来验证完全仲裁。

(4) 预防篡改：修改时系统怎样保护访问监控器，包括它的保护体系？

受到干扰时，Windows 遇到与 UNIX 一样的问题。首先，保护体系是无条件的，所以它可能受到任何运行进程的干扰。非信任的用户进程可能随意修改它们的用户数据的访问权限，所以加强用户数据的安全目标是不可能的。因为用户常常以管理员身份登录来缓解系统管理，保护体系的各个方面都可能被修改。第二，内核本身有受限保护。与 UNIX 相同，一个 Windows 内核可能通过内核模块被修改。在 Microsoft Vista 中，一个代码签名过程可能被用来确定内核模块（比如，签名者，不一定是模块的写入方）的证明者。当然，管理员（一般是终端用户）必须能够确定签名者的可信赖性。取决于用户决策的安全程序常常会失败，因为用户常常忽略这些决策的安全含义。而且，与 UNIX 类似，Windows 内核也不能按照系统要求定义保护。

（5）预防篡改：系统的保护体系保护可信计算基础程序吗？

Windows 系统的可信计算基（TCB）与 UNIX 差不多。几乎任何程序都可能是 Windows 的 TCB 的一部分，而且任何运行这些程序的进程都可以修改其他的 TCB 程序使其无效。

与 UNIX 一样，任何被盗用的 TCB 进程都可以修改保护体系使得系统全目标的强化部分无效，并且通过提供给 TCB 进程访问内核状态的各种接口修改 Windows 内核本身。

不同于 UNIX，Windows 提供 API 篡改其他进程（UNIX 没有这种功能）。举一个例子，Windows 提供 CreateRemoteThread 函数，这个函数可以在其他进程中开始一个线程。Windows 也提供写入进程存储器的函数（OpenProcess 和 WriteProcess-Memory），所以一个进程也可以在另一个进程开始一个线程之前向那个进程中写入所要求的代码。然而所有这些操作都要求一定的访问其他进程的权限，通常需要改变调试一个进程所必需的权限（通过 AdjustTokenPrivileges 函数）。这些特权一般仅仅对 SID 相同的进程可用，必须证实这些特权不被滥用，从而保证我们的 TCB 不被篡改。

（6）可证明：系统的可信计算基础的正确性的根据是什么？

对 UNIX 而言，任何正确性的根据都是不规范的。Windows 也有一个无限 TCB 和可扩展的内核系统，它们防止任何有效的形式化证明。

（7）可证明：保护体系加强了系统的安全目标吗？

一般的 Windows 模型可以对访问权限的组合进行具体说明，但是这种系统中没有定义任何特殊的安全目标。这样，一个系统是否安全是不可能知道的，因为这个模型比 UNIX 模型更复杂，它可以随意扩展，这使得安全的验证也更困难。

3.7.4　Windows 漏洞

由于 Windows 自身的局限性，Windows 有和 UNIX 系统相同的漏洞。这里将对 Windows 系统特有的以及在 Windows 系统中更为深奥的一些漏洞进行描述。

1. Windows 注册表

Windows 注册表是一个存储所有程序的数据的完整的分层数据库。当一个新的应用程序被下载时，注册表可能被更新为特定的应用程序，比如安全敏感信息（访问库的路径以及为应用程序下载的可执行文件）。每个注册项都有一个与之相连的限制访问的安全上下文，所以一般不能有效地使用这些限制。比如，AOL 的标准配置添加了一个注册项，该项具

体说明了一个将被 AOL 软件下载的 Windows 库文件的名字(如 DLL)。然而,由于设置了许可,所以任何用户都能够写入项。

注册表的这个使用是常见的,因为卖家必须保证他们的软件在被下载后能够运行。一般地,如果用户下载了某个新购买的软件,但这个软件因为不能访问它的重要的库文件而导致不能正确运行,那么用户会不高兴。因为应用程序的卖家不知道 Windows 系统被管理的特定的方式,他们必须得到一个许可来保证用户能够运行软件。如果注册表项后来被攻击者用来危害 Windows 系统,那么这不是应用程序卖家的问题。

2. 管理员用户

可以注意到 Windows 安全评估中用户习惯以管理员身份运行系统,或者至少使用管理特权。这个原因与授给注册项的访问权是一样的:用户也想确认他们能够使用使系统运行的必需的函数。如果用户下载某个电脑游戏,那么用户需要特权来安装这个游戏,还可能需要特权来运行设备密集型游戏程序。最后一件用户希望的事情是必须找出为什么这个游戏不能玩,所以要让所有的特权围绕着这个问题工作。

UNIX 系统的用户更多的是有经验的计算机用户,他们知道安装软件与正常操作计算机的不同。因此,UNIX 中 root 用户与 sodu 操作的使用更有效。

3. 默认许可

与用户和软件卖家一样,Windows 调度也开始于完全许可和功能启用。这导致了著名的"红色代码"(CodeRed)蠕虫,该蠕虫攻击 Microsoft IIS Web 服务器的 SQL 服务器部分。许多运行 IIS 的用户没有运行 SQL 服务器,他们甚至不知道 SQL 服务器在 IIS 系统中是默认许可的。但是在这些平静的时刻,IIS Web 服务器能够运行所有的软件,所有攻击者能够发送恶意请求给任何系统的 SQL 服务器,触发缓冲区溢出是这个蠕虫发起的基础。现在 IIS 的后续版本已经"锁定"了这个功能,这样软件必须进行手动操作才能访问。

3.8 可信操作系统

如果信赖一个系统的安全性,就认为这个系统是可信的。如果一个可信系统不能提供想要的安全,那么这个系统的安全就是失败的。

信任与安全是不同的,信任隐含着信赖,即信任是二元选择——要么信任要么不信任。安全则是对安全机制有效性的判断。安全的判断与规定的策略和声明相关。然而,安全依赖于信任,因为一个可信部分不能提供期望的安全便会破坏系统的安全。理想情况下,仅信任安全系统,并且所有的信任关系是毫不隐瞒的。

因为可信系统可以相信它的安全,所以不可信的系统就不信赖它的安全。因此,如果所有的不可信系统都受到破坏,那么系统安全是不受影响的。这隐含着只有可信系统能够破坏安全,这个结论很重要。

操作系统调解主体(用户)与客体(资源)之间的关系。可信操作系统必须安全地提高分离、内存保护以及访问控制,即可信操作系统必须决定哪个客体被保护,怎样保护,哪个主体允许这样做。

任何好的安全原理一般都包括最小特权(比如低水位线原理)、简单、开放设计、完全仲裁、白名单(相对的是黑名单)、分离以及易用性。可信操作系统应该能够安全提供所有这些功能。然而,这也导致了大部分可信操作系统的复杂性和安全性差。

3.8.1 可信操作系统的概念

可信操作系统是安全操作系统的进一步发展。安全操作系统是可信操作系统的基础。没有安全操作系统就没有可信操作系统。可信操作系统不是主观臆造的,是在安全操作系统的基础上发展起来。安全操作系统中的安全政策、体系结构及其实现方式仍然适合可信操作系统,但是,它们之间也存在一些不同之处。直观地说,可信操作系统研究的是如何为用户提供一个可信的计算环境,而安全操作系统研究的是如何为用户提供一个基础安全平台。

提到可信,人们很容易想到美国国防部的 TCSEC 评估准则中定义的 TCB 的概念。这种 TCB 安全核的概念对于操作系统和其上建设的数据库管理系统 DBMS 以及应用系统都是有意义和有效的。在操作系统的安全观念中,也按照角色(例如管理员、超级用户、普通用户等)给予一定的引用系统服务的权限。也就是说,在 TCSEC 中提出的可信概念实际上是一种特权概念。只要是系统的管理员或超级用户,就认为是可信的。

与 TCB 比较起来,可信操作系统具有以下三个特点。

(1) 主体行为的可信性:这里主体行为的可信性是指考察行为的预期性,也就是说,不可信行为对于从事某项工作来说不是必要的或预期的。

(2) 客体内容的保密性、完整性和可信性:客体内容的机密性是指防止信息泄露给未授权的用户;而完整性是指防止未授权用户对客体内容的修改;客体内容的可信性就是客体内容的真实性。

(3) 自身的完整性:自身的完整性就是操作系统自己不被非法篡改。

总之,可信操作操作系统必须确保用户身份、主体行为的可信性,这是对使用者的信任;确保信息存储、处理、传输的机密性、完整性、真实性,确保服务及应用程序的完整性,体现了客体内容的可信;确保密钥操作和存储的安全;确保系统具有免疫能力,从根本上防止病毒和黑客。

3.8.2 MAC、DAC 及其他

如图 3-10 所示,任何操作系统都必须提高一定程度的分离,内存保护和访问控制。另一方面,可信操作系统,如图 3-11 所示,必须确实提供另外的安全,包括强访问控制、自主访问控制、客体重用保护、完全仲裁、可信路径以及日志。

强制访问控制(MAC)是指不是由客体的所有者控制访问。比如,Alice 不对谁享有 TOP SECRET 情况权限做出决定,所以她不能完全控制同一级别文件的访问权。直接访问控制(DAC)是指访问由客体的所有者决定的访问控制。比如,UNIX 中的文件保护,文件所有者控制读、写以及执行权。

如果 DAC 与 MAC 都用于一个客体,MAC 更有利。假设 Alice 拥有 TOP SECRET 安全级别的文件,Alice 可以设置 DAC(因为她拥有该文件),但是,不管这些设置如何,仅有 SECRET 权限的 Bob 不能访问这个文件,因为他不满足 MAC 的要求。

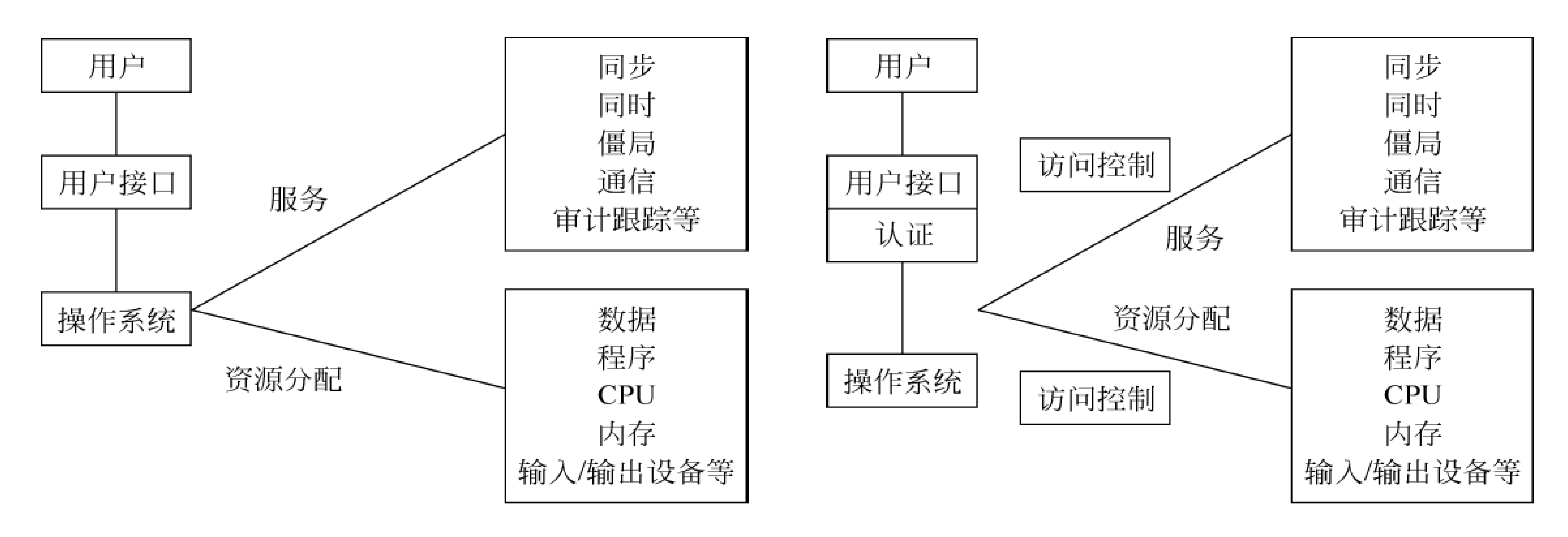


图 3-10　操作系统概述　　**图 3-11　可信操作系统概况**

可信操作系统还必须保护信息从一个用户泄漏给另一个用户。任何操作系统都使用内存保护和访问控制的形式来避免这类信息泄露。但是一般希望从可信操作系统中得到更多。举一个例子，当可信操作系统为文件分配空间时，相同空间可能优先被一个不同的用户进程使用。如果操作系统没有相应的预防措施，保存先前进程的位就可能被访问，这就造成信息泄露。可信操作系统应该采取措施来避免此类情况的发生。

与此相关的问题是"剩磁(magnetic remanence)"，先前存储的模糊图像有时可能被读，即使这个空间被新的数据重写了。为了减少这种现象发生的概率，美国国防部(DoD)设置的准则是在允许其他进程访问此空间之前用不同的位形式反复重写内存。

3.8.3　可信路径

计算机系统中，用户在一般情况下并不直接与内核打交道，中间还有一层应用层作为接口，在用户与内核之间相互作用。虽然这种设计能够保护内核不会被用户肆意修改和窥测，但也带来了安全问题。由于应用层并不能完全信任，因此在操作系统安全功能中往往会提供可信路径这一功能。可信路径(trusted path)是指通过应用层，在用户与内核之间开辟一条直接的可信任的交互通道。

当输入密码进行即时登录时，密码会怎样完全取决于系统中运行的软件。那怎样才能确定软件没有进行恶意活动，比如将密码写入文件然后用邮件发送给 Turdy？这就是可信路径问题。Ross Anderson 是这样说的："我不知道怎样才能相信，即使是我在自己的 PC 上完成的数字签名，我从事安全工作已经超过十五年。在演示软件和签名软件间严格检查所有的软件的路径已经让我失去了耐心。"理想地说，可信操作系统应该提供强信任的可信路径。

攻击者获取非法访问权限的方法之一就是"欺骗"用户，使他们误认为与之进行交互的是一个合法的安全系统。而实际上，用户的键入操作完全被攻击者截获并分析。举一个例子，一个恶意的"骗子"在用户和合法系统之间设置了一个伪造的用户名和口令系统，然后他就可以正常登录了。所以，诸如设置口令，或是更改权限这些重要的操作就需要一个可靠的、不会出现上述问题的环境来保证其安全性。于是就出现了可信路径这一概念。它可以保证用户的重要信息只能由合法的接受者来接受。在一些可信系统中，用户是通过一串唯一的关键字来激活一个可信路径的；也有其他的一些可信系统，规定只有在系统启动时，即

在安全执行编码已经运行且还没有任何处理动作之前，才可以做与安全相关的信息的更改，以此来保证安全性。

操作系统必须对登录安全相关的事件负责。为了检测攻击和事后分析，这些信息很重要。登录并不像它看起来那么简单，有时候不能进行正确登录。但是，如果登录太多，可能会让那些必须检查数据的人筋疲力尽，也可能让必须在大量数据中寻找相关信息的自动系统筋疲力尽。比如，应该登录正确的密码吗？如果这样，那么大部分密码会出现在日志文件中，日志文件自身也会受到安全检测；否则，进程中的口令猜测攻击就很难被检测到。

3.8.4　可信计算基

内核是操作系统级别最低的部分，负责同步、内部进程通信、信息传递、中断处理等。安全内核是内核中处理安全的那部分，即安全内核包含在内核之中。

为什么有个安全内核呢？因为所有的访问必须经过内核，它是访问控制的理想位置。它可能也是在某处设置安全监测功能的很好的实践。将所有的这样的功能放在同一个位置，安全功能更容易检测和修改。

推理监测是安全内核的一部分，它处理访问控制。推理监测调解主体和客体之间的所有访问，如图 3-12 所示。安全内核的重要部分是防篡改，理想地说，它必须可分析、小巧、简单，因为这个层次出错会危害整个系统的安全。

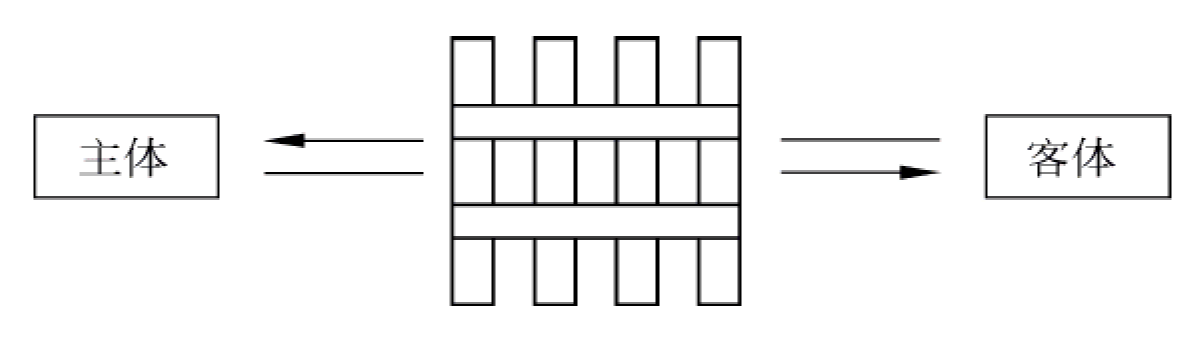


图 3-12　推理监测

为了维护每个域的保密性和完整性，可信计算基(TCB)监控以下四个基本活动：

(1) 进程激活。在进程的环境中，激活和撤离进程的频率是很高的。从一个进程切换到另一个进程需要完全改变注册表，重置映射，并改变文件访问表、进程状态信息和其他一些指针，其中大部分都是秘密级别的信息。

(2) 执行域切换。运行在某一域的进程经常会激活其他域中的进程以获取更高秘密级的数据或服务。

(3) 存储器保护。因为每一个域中都包含存储在内存中的代码和数据，TCB 就必须监控内存访问来保证每一个域的完整性和保密性。

(4) I/O 操作。在一些系统中，软件涉及 I/O 操作，外层输入与最内层的 I/O 设备会连接起来，这样，I/O 操作要穿过所有的域，所以必须被监控。

操作系统中的安全操作可能会出现在许多地方。理想地说，首先设计安全内核，然后在它旁边建立操作系统。不幸的是，实际往往是相反的，因为安全是一个事后想法而不是先前的设计目标。尽管如此，可信操作系统却是将安全作为初始目标。Honeywell 开发的可信操作系统 SCOMP 就是这样的一个例子。SCOMP 为了达到简单性和可分析性，安全内核不到 10000 行代码，与此相反，Windows XP 用了超过 40000000 行的代码。

理想地说，TCB 应该在一个可确认的层集合所有的安全功能。图 3-13 中的 TCB 是一

个设计简单的例子，因为安全特征遍布于整个操作系统。任何对安全特征的改变都可能对其他操作系统功能产生影响，而且单个安全操作很难进行分析，尤其是与其他安全部分进行通信的时候。

图 3-14 所示的 TCB 设计就很好，因为所有的安全功能都集中在定义的安全内核上。在这个设计中，任何安全功能改变引起的安全影响都能够通过研究安全内核的影响来分析，而且，攻击者在高层次破坏操作系统不会引起 TCB 操作的失败。

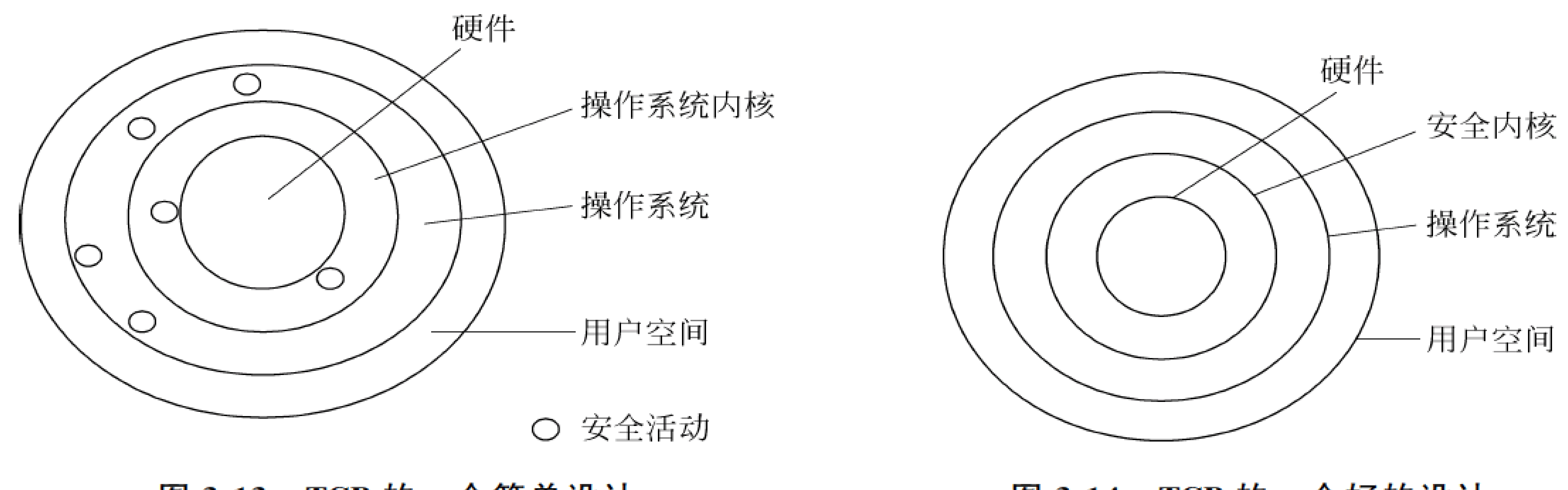


图 3-13 TCB 的一个简单设计　　**图 3-14 TCB 的一个好的设计**

总的来说，可信隐含着信赖，TCB 是可信赖的操作系统安全的所有方面。如果 TCB 之外的任何东西被破坏，则仍然有一个可信系统；但是如果 TCB 内部被破坏，系统的安全就被破坏了。

习 题 3

1. 有几种分离方式可以用来保护操作系统？它们的优缺点是什么？
2. 段式保护与页式保护相比有什么优缺点？如何解决这些缺点？
3. UNIX 系统中都使用了什么机制来保护文件的安全性？
4. 列举一些用户认证方式，这些认证方式的优缺点各是什么？
5. UNIX 系统与 Windows 系统的安全保护体制有哪些不同？
6. 什么是可信操作系统？
7. 解释一下可信路径和可信计算基的概念。

第 4 章　数据库安全

计算机技术的飞速发展使得数据库的应用不断深入到各个领域，但随之而来产生了数据的安全问题。各种应用系统的数据库中大量数据的安全问题、敏感数据的防窃取和防篡改问题，越来越引起人们的高度重视。数据库安全问题的研究始于 20 世纪 70 年代，美国军方率先发起对多级安全数据库管理系统的研究。

4.1　数据库安全威胁

原则上，凡是造成对数据库中存储数据的非授权的读取、写入、添加、删除、修改等，都属于对数据库的数据安全造成了破坏。另一方面，当正常业务需要访问数据库，但授权用户不能正常得到数据库的数据服务时，也称之为对数据库的安全构成了破坏。很明显，这两种情况都会对数据库的合法用户的权益造成侵犯，或者是信息的被窃取，或者是由于信息的破坏而形成提供错误信息的服务，或者是干脆拒绝提供服务。

对数据库安全的威胁或侵犯大致可以分为自然灾害、人为疏忽、恶意破坏、犯罪行为、隐私侵害。

图 4-1 汇总了各种数据库安全威胁及其来源。

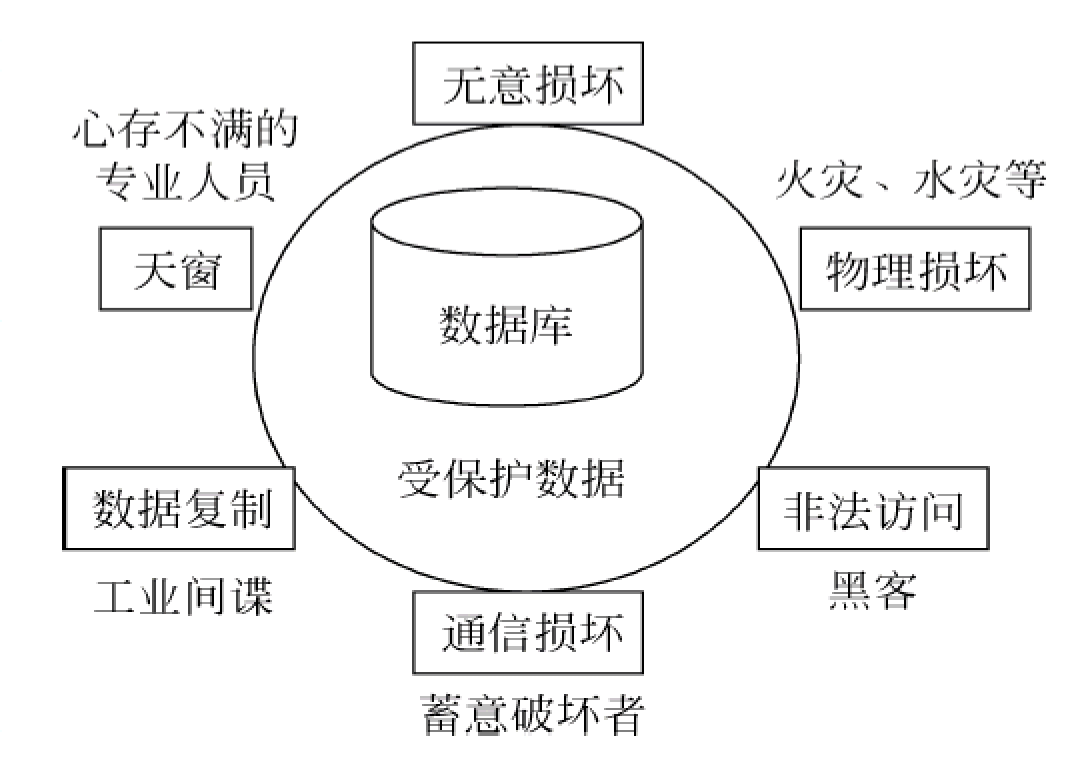


图 4-1　数据库安全威胁

4.2　数据库安全需求

数据库的基本安全需求主要包括以下几个方面。

（1）数据库完整性。数据库完整性包括物理数据库完整性和逻辑数据库完整性两部分。

- 物理数据库完整性。数据库的数据不受物理问题（比如停电）的影响，数据库如果因为灾难而被毁坏是可以被重建的。
- 逻辑数据库完整性。数据库的结构是受到保护的。因为数据库的逻辑完整性，修改一个域的值不会影响其他的域。

如果数据库作为中心数据仓库，用户必须相信数据值的正确性，这就表明数据库管理员必须保证更新只能由授权个体进行，这同时说明数据必须防止被破坏，无论是外部非法程序还是火灾、停电等外部因素。这些因素都会影响数据库的完整性：整个数据库被破坏（存储

介质被破坏)或者单个数据条目不能读。

重建数据库有时很重要,比如,突然停电时,银行用户可能正在交易中,学生可能正在注册他们的在线课堂,此时,我们希望将系统恢复到稳定点,而不需要用户重新进行他们最近完成的交易。为了处理这种问题,DBMS(Database Management System,数据库管理系统)必须保存交易日志。举一个例子,假设银行系统就是这样设计的,每次交易发生时,在日志(电子或纸质都可)中产生消息。一旦系统失败,系统恢复到数据库的备份,从而获得准确的账户余额,重新处理日志后面的交易。

(2) 元素完整性。每个元素包含的数据都是正确的。

数据库元素的完整性是指它们的正确性和准确性。授权用户必须输入正确的数据到数据库,然而,用户或者程序在收集数据、计算结果或者输入数值的时候可能会发生错误。因此,DBMS 有时采用特殊的方法来帮助发现错误并改正错误。

改正措施有三种:第一种完整性措施是在 DBMS 中采用域检查,在一些位置测试值是否合适。一个域可能要求是数字、大写字母或者可接受的字符串。通过检查确认某个值错误是因为不在限定的范围内或者不大于另外两个域的值的和,这些检查防止了简单的输入数据的错误。

第二种完整性措施是访问控制。不妨假设没有数据库,那么数据文件可能包含多个资源的数据,冗余数据可能存储在不同的地方。比如,一个学生的家庭住址可能存储在不同的学校文件中:班级注册、书店、财政资助中心。事实上学生并不知道每个不同的地点的文件都有地址信息。如果学生从一个地方搬到另一个地方,每个单独的文件都需要更正。没有数据库,数据的完整性就可能遭到破坏。首先,可能某时刻有的数据文件存的是旧地址(还没更新),而有的文件存的则是新地址(已经更新了);其次,数据文件可能被不正确地修改,从而导致域的信息值不正确;第三,可能有学生不知道的文件存在,那么学生不知道通知文件所有者更新地址信息,这些问题都可以通过数据库系统解决。他们允许在中心资源收集和控制这个数据,确保用户拥有正确的地址。

然而,集中化也并非容易。谁享有这个共享的中心文件?谁授权更新那个元素?如果两个人产生了相互矛盾的修改怎么办?如果修改没有按照顺序怎么办?检查到重复记录怎么办?发现重复采取什么措施?这些问题都需要数据库管理员解决。

第三种完整性措施是保存数据库的修改日志。修改日志列出了数据库的每次修改的信息;它包含原始值和修改后的值。使用这个日志,数据库管理员能够撤销所有错误的修改。

(3) 可审计性。能够对数据库中元素的访问和修改进行跟踪。

在一些应用中,在数据库中产生所有访问的审计记录是很有利的,有助于保持数据库的完整性。比如,单一的访问不会泄漏受保护的数据,但是将连续的几个访问组合在一起就可能泄漏受保护数据,审计跟踪能够识别出用户已经获得的信息,从而决定是否告诉该用户更多的信息。

颗粒度不利于审计的进行。操作系统中的审计事件是诸如打开文件、请求程序之类的活动;它们不像“写记录 3”或“执行指令”那样具体。为了有利于保护完整性,数据库审计跟踪应该包括记录、域以及事件元素级别的访问。大部分数据库应用中这些细节是禁止的。

此外,记录被访问但是没有报告给用户是很有可能的(比如当用户执行一个选择操作

时)，此类访问记录或元素但没有将接受到的数据返回给用户的情形称为窗口问题(pass-through problem)。注意，不访问数据而确定一些元素的值是完全可能的，举一个例子，当你知道一个组中员工只有一人时，你也就知道了这个组的员工的平均收入。因此，直接访问记录的日志可能夸大或缩小用户实际知道的信息。

(4) 访问控制。用户只能访问获得访问许可的数据，而且不同用户的访问模式(读、写等)不同。

用户访问特权常常从逻辑上分离数据库，比如，所有的用户可能都被授权访问一般的数据，但是只有人事部能够获得工资数据，只有销售部能够获得销售数据。此时数据库就显得很有用了，因为它们集中存储和保护数据，限制访问是这种集中化的职责，同时也是其优点。

数据库管理员从视图、关联、域、记录甚至元素级别对谁应该访问哪个数据进行划分。DBMS 必须加强这个策略，对所有的访问数据赋予访问权限或禁止访问。此外，访问模式可以是多样的。用户或程序可能有权读、修改、删除或者附加一个值、添加或删除整个域或记录，或者重组整个数据库。

表面上看，数据库的访问控制就像操作系统的访问控制，或者计算机系统的其他部分。然而，数据库问题更为复杂。操作系统的客体(比如文件)是无关系的条目，而记录、域以及元素都是相关的。尽管用户不能通过读其他文件来确定某文件的内容，但是用户可能通过读其他数据元素来确定某个数据元素。从其他数据值获得某个数据值的问题就是推理，后面将详细讲解。

不直接访问安全客体本身而通过推理访问数据是很值得注意的。限制推理意味着禁止某些路径来防止可能的推理。然而，限制访问控制推理也限制了那些不打算访问非授权值的用户的请求。而且，试图对可能存在的不可接受的推理进行检查访问的请求实际上可能降低 DBMS 的性能。

最后，操作系统客体和数据库客体的颗粒度是不同的。上百个文件的一个访问控制列表的执行比有上百个文件(每个文件可能又包含一百个子文件)的数据库的访问控制列表要容易得多。粒度影响处理效率。

(5) 用户认证。为了进行一定数据的审计跟踪和访问许可，每个用户都要进行身份认证。

DBMS 能够请求用户认证，比如，DBMS 可能强行要求用户通过具体的口令和当天时间检查，这个认证补充了操作系统的认证。一般地，DBMS 作为应用程序在操作系统之上执行。这个系统设计意味着没有从操作系统到 DBMS 的可信路径，所以 DBMS 必须怀疑它收到的任何数据，包括用户认证。这样，DBMS 被强制执行它自己的认证。

(6) 可用性。用户一般能够访问数据库，以及所有授权的数据。

DBMS 有程序和系统的特征，它是一个使用其他硬件和软件的资源，但是对很多用户而言它是唯一的运行程序。用户常常不重视 DBMS，把它作为一个处理特殊任务的重要工具。但是当系统不能为其他用户服务时，或者需要修理或升级时，用户便认识到 DBMS 的可用性。举一个例子，两个用户可能请求相同的记录，DBMS 必须做出仲裁，一个用户必须稍后才能访问，或者 DBMS 可能拒绝未受到保护的数据以避免下载受保护的数据，这将使得请求用户不高兴。这类问题导致了 DBMS 可用性需求的提出。

4.3　可靠性与完整性

数据库从许多资源聚集数据，用户期望 DBMS 以可靠的方式提供数据的访问。当软件工程师说软件是可靠的，他们是说软件可以长时间运行而不发生问题。用户肯定期望 DBMS 是可靠的，因为数据常常是商务和组织的重要部分。而且，用户让 DBMS 保管他们的数据，是希望它保护数据，防止数据丢失或损坏。

DBMS 通过多种方式防止丢失和损坏，这里所要考虑的不是绝对保护，因为没有一种保护能够阻止授权用户不小心访问允许的但不正确的值。

数据库关心可靠性和完整性可以从以下三个方面看出。

（1）数据库完整性：数据库是作为一个整体来防止破坏，比如磁盘驱动的失败、管理数据库索引的破坏。操作系统的完整性控制和恢复程序可以处理这类问题。

（2）元素完整性：只有授权用户能够对具体的数据元素的值进行写或者修改。合适的访问控制保护数据库防止被授权用户破坏。

（3）元素准确性：只有正确的数值能够被写入数据库。检查元素的值有助于防止不合适的值的插入，而且，限制条件也能够检查不正确的数据。

4.4　敏感数据

敏感数据是一个相对概念，不被用户直接访问到的数据对该用户而言就是敏感数据。数据的不可访问通常可以分为两种情况，一种是由于工作需要的划分，另一种则是由于安全等级的限制。前者出现在基于授权的安全模型中，未授权以及反授权的数据对某用户而言就是敏感的，并且二者的敏感性也不同；后者出现在基于标记的多级安全模型中，高安全级别的数据相对于低安全许可的用户显然是敏感的。相应地，在混合的安全模型中，用户推理得到的数据可以分为两种情况，一种是数据的安全级别并不高于该用户，但由于自主访问控制的限制使得该用户无法直接访问到；另一种是数据的安全级别高于该用户，因为强制访问控制而使用户无法直接访问到。这些都是推理控制所必须面对的。

可以通过敏感级别（Sensitive Level，SL）来统一刻画数据的敏感性。在授权控制机制下，授权客体的敏感级别低于未授权客体，更低于反授权客体（这类客体可以把自己所拥有的权限反授给比自己安全等级高的客体）；而在强制控制机制下，低安全级别客体的敏感级别低于高安全级别客体。这样，对于给定的用户，如若推理得到的数据在敏感级别上高于其可以正常访问的数据，则该推理结果即为敏感的。考虑到叙述的方便，后文如无说明仅将敏感级别分为两级：High 和 Low。

作为推理控制的对象，敏感数据必须存储于数据库中，在安全粒度上对敏感数据做出任何限制都会影响系统的实际应用。因此，根据安全系统的设计需要，敏感数据应该既可以是单个数据项，也可以是数据项的组合；既可以是整个关系中的所有元组，也可以是满足条件的部分元组。关系数据库中，属性是数据的最小结构单位，敏感数据要存储在数据库中必然

依附于属性,因此可以通过属性并以下面的方式统一描述各种安全粒度的敏感数据。

选择：给定关系实例 r 上的一个选择可以表示为 $r(F)$,其中 F 是由通过逻辑运算符连接的多个形如 $a\theta b$ 的算术表达式构成的一个逻辑表达式,其中 θ 为比较运算符,a 是 r 中的属性名,b 是对应的常量。用 $F=1$ 表示无选择条件。

选择的结果是 r 中使得 F 为真的元组子集。

查询：给定关系实例 r 上的一个查询可以表示为 $Q(obj)$：con,其中查询条件 con 是 r 上选择的有限集合,查询对象 obj 是 r 的属性子集。通过简单的 SQL 转化,查询即可精确定位数据库中的数据,因此使用查询也可以形式化定义数据库中各种粒度下的敏感数据。

敏感数据：相对于某用户的一条敏感数据是数据库 D 中由查询 $Q(obj)$：con 确定的、其无法直接访问的数据,并且称 $|obj|=1$ 时的敏感数据为单一敏感数据,而称 $|obj|>1$ 时的为组合敏感数据。

单一敏感数据具备敏感一致性,即所有组成元素的敏感级别相同；对于组合敏感数据,考虑到数据库的可用性,其组成元素应可以有与组合整体不同的敏感级别。由此约定：①单一敏感数据中,任何数据的泄露等同于整体的泄露；②组合敏感数据中,只有在所有组成元素泄露之后才会导致整体的泄露。相应地,单一敏感数据的推理控制必须完全隐藏其涵盖的所有数据,而组合敏感数据的推理控制只需要隐藏其组成元素之一涵盖的数据。

4.5　多级数据库

到目前为止,我们都是将数据分为两类：敏感数据和不敏感数据。有些数据条目比其他的更敏感,但是我们仅允许“是”、“否”访问。这就是说敏感性是一个属性函数,数据出现在纵列。表 4-1 给出了一个这种模型,其中加粗的两列认为是敏感的。事实上,敏感性不仅仅是由属性决定的。

表 4-1　属性级别敏感性

名　字	部　门	薪　水	电　话	职　位
Rogers	培训	**43800**	4-5067	**A2**
Jenkins	科研	**62900**	6-4281	**D4**
Poling	培训	**38200**	4-4501	**B1**
Garland	客服	**54600**	6-6600	**A4**
Hilten	客服	**44500**	4-5351	**B1**
Davis	管理	**51400**	4-9505	**A3**

4.5.1　不同安全的实例

假设一个数据库包含美国政府支出的数据。这些支出中有的是用于日常用品,不是敏感信息；有的是工资支出,需要保密,个人的薪水是敏感数据,但是总的支出不是敏感的。一些军事操作的支出则更敏感,比如美国用于导弹的支出总额(这是不公开的)。还有一些操作仅能被少数人知道,这些操作的花费额也显得尤为敏感。

表 4-1 列出了员工信息。事实上可能会出现这样的情况，Davis 是某个特殊项目雇佣的临时工，她的所有记录与其他员工的敏感性不同；或者显示的 Garland 的电话是她的私线，并不对外公开。根据上面的情况，表 4-2 完善了数据的敏感性。

表 4-2 数据和属性敏感性

名字	部门	薪水	电话	职位
Rogers	培训	**43800**	4-5067	**A2**
Jenkins	科研	**62900**	6-4281	**D4**
Poling	培训	**38200**	4-4501	**B1**
Garland	客服	**54600**	**6-6600**	**A4**
Hilten	客服	**44500**	4-5351	**B1**
Davis	**管理**	**51400**	**4-9505**	**A3**

表 4-2 显示了数据库安全的三个特点：①单个元素的安全等级可能与相同记录的其他元素（或者相同属性的其他值）不同，即一个元素的安全等级可能与相同行或列的其他元素不同，这说明安全需要对单个元素实现；②在一些条件下两级敏感或不敏感是不够的，需要更多的安全等级，这些等级可能代表着允许程度的排列（可能重叠），安全等级形成一个点阵；③数据库中总和、数量、群值的安全性可能与单个元素不同，它们的安全级别可能高于或低于单个元素的安全级别。

这三条原理形成了多级数据库的安全模型，但与军事模型的安全性不同，军事模型中客体的敏感性定义为 n 等级，并按照种类划分成区间。

4.5.2 粒度

试想如果将原来用于纸质文件的军事分类模型用于计算机，那么分类并跟踪一张纸、一个纸质文件、一个计算机文件、一个程序或进程都是相当容易的。对单个数据条目进行分类是完全不同的。

由于一些原因，一张完整的纸分为一个等级，即使一些单词（诸如 and、the、of 之类）在文中没有多大意义，而核心单词像 Manhattan project 之类可能是在文中的敏感的。定义数据库中每个值的敏感性与文件中的每个单词使用的敏感性是类似的，但是更为复杂。单词 Manhattan 和 project 本身并不敏感，但是它们组合起来成了敏感的核心单词 Manhattan project。数据库中也有同样的情况，因此，不仅数据库中的每个元素有一个不同的敏感度，元素的每个联系也有一个敏感度。而且，元素之间的联系比它的任何一个元素都敏感。

为了让数据库中的每个值都有一个敏感级，需要什么呢？第一，需要一个访问控制策略来规定哪个用户有权访问什么数据。一般为了实现这个策略，将每个数据条目都注上标记以显示访问限制。第二，需要一个方法，保证数值没有被未授权的用户修改。这两个要求满足了机密性和完整性。

多级数据库的控制项的粒度小，所以安全管理实现难度也很大。可以采用分区的机制，把一个大的数据库分成几个分离开的数据库，每个数据库都有自己的敏感级别。另外还可以采用加密机制，如果敏感数据被加密了，即使一个用户偶然获得了这些数据也无法解码还原数据。

4.5.3　安全问题

1. 完整性

即使所有元素的敏感度都相同的单级数据库，完整性也是很棘手的问题。多级数据库中，完整性的实现更复杂也更难，因为访问控制的 * 属性，读高级别数据的进程不允许写入一个低级别的文件。然而，用于数据库时这个原则是说高级别的用户应该不能写一个低级别数据元素。当 DBMS 必须读数据库中的所有记录，并因为备份、浏览数据库回答请求、根据用户的进程需要重组数据库、或者更新数据库的所有记录等目的要写入新记录时，这个问题更加严重。

当人们遇到这个问题时，他们通过信任和常识来处理。对敏感信息有访问权限的人很小心地避免将它们传递给不了解的个体。在计算机系统中，有两个选择：一是进程在高级别安全等级时不能写入低级别进程，二是进程必须是"被信任的进程"，计算机与人有相等的安全等级。

2. 机密性

用户相信数据库提供正确的信息，即数据是符合条件的、正确的。保护机密性的一些方法可能导致数据的一些修改。尽管这些置换不影响统计分析，但是为了响应两个不同形式的请求，它们会产生代表同一个潜在数据值的两个不同的答案。在多级问题中，两个不同的用户操作两个不同的安全级别，对相同的请求可能产生两个不同的答案。为了保护机密性，牺牲了精确性。

加强机密性也导致了不知道的冗余。假设一个人事专家工作的安全级别是访问许可。专家指导 Bob Hill 为公司工作，但是，Bob 的记录没有出现在退休金名册中。专家确信这个遗漏是一个错误，并为 Bob 创建了一个记录。

没有 Bob 记录的原因是 Bob 是一个秘密代理，公司中他的招聘部门不公开。Bob 的记录在一个文件中，但是因为他的特殊地位，他的记录人事专家无法访问到。DBMS 不能拒绝人事专家的记录，因为这样做就表示这里存在一个敏感度很高、人事专家无法看到的记录。新纪录的创建意味着 Bob Hill 有两个记录：一个是敏感的，另一个则不是，见表 4-3。这种情况称为多实例(polyinstantiation)，也就是说一个记录能够多次出现，但每次的机密等级都不同。

表 4-3　多实例记录

姓　　名	敏　感　度	任　　务	地　　点
Hill, Bob	C	**Program Mgr**	London
Hill, Bob	TS	**Secret Agent**	South Bend

这个问题很严重，因为 Bob Hill 是一个常见的名字，数据库中可能有两个不同的人有相同的名字。这样，仅仅为了重复的名字浏览数据库(从高敏感度)不是找到低安全级别权限的人无意输入记录的很好的方法。

我们可能会找到其他原因，与敏感级别无关，那将导致多实例。比如，Mark Thyme 在 Acme 公司工作了 30 年退休了，他现在从 Acme 领取退休金，所以他作为一个退休人员出

现在人事记录中。但是 Mark 不愿意在家中,他作为一个临时工被重新雇佣,这个新工作产生了一个 Mark 的新的人事记录。每个都是合法的聘用记录,我们想要减少多实例,必须消除这样的合法记录。

4.6 推理控制

数据库系统提供的访问控制机制作为一种安全手段能加强对敏感信息的保护。在这种保护机制下,敏感信息的访问是通过访问预先设定的数据库接口来实现的。但这种重要而且必不可少的访问控制机制对信息的安全保护通常是不够的,因为非法用户或内部人员可以绕过访问控制机制达到颠覆敏感信息安全的目的,这也就是通常所说的推理访问,以间接方式导致敏感信息泄漏的推理访问是安全数据库中所存在的重大安全隐患。多级安全数据库、统计数据库、一般数据库、数据挖掘等都不同程度地存在推理导致敏感信息泄漏的问题,因此推理访问控制对于保证数据的安全具有非常重要的意义。

4.6.1 推理控制的基本概念

虽然访问控制模型防范了敏感信息以直接访问的方式被泄露,但仍然存在通过推理通道以间接访问的方式来颠覆敏感信息的可能。如果用户不通过访问敏感信息而推理得到了敏感信息,数据库管理系统中的任何限制措施也就都失去了意义。同样,也没有办法知道数据库管理系统中是否存在这样的非限制信息,利用这些非限制信息可以推出数据库管理系统之外的敏感信息。当敏感信息通过访问非敏感信息或元数据而泄露出去时,就说安全数据库中存在推理问题。这样的过程可以表示为如图 4-2 所示。图中的情景表明,用户从数据库管理系统返回的查询结果信息中推理得到了额外的数据信息。

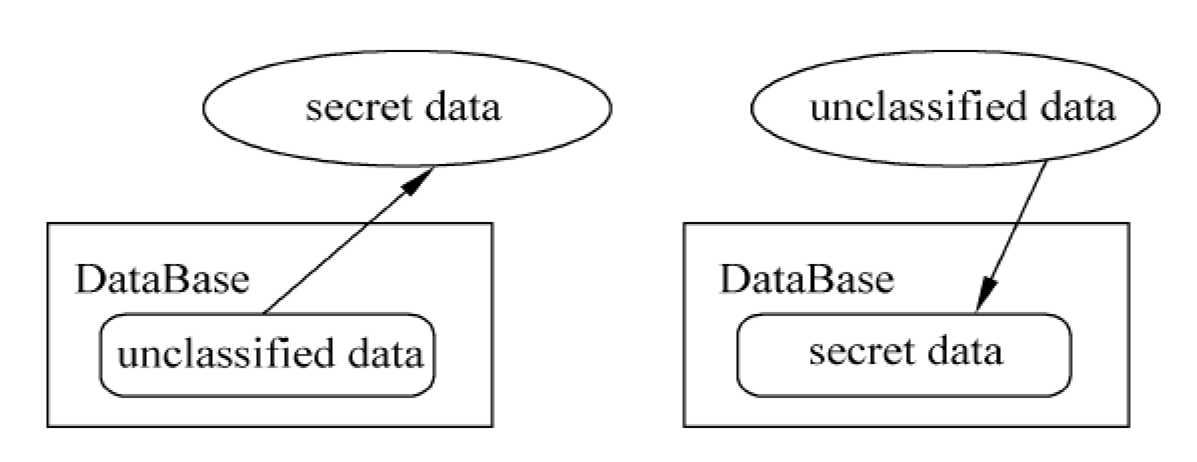


图 4-2 发生推理访问控制的情景

这里所说的元数据可能包括数据库约束,如数据信赖、完整性约束或者用户的背景知识、相互关联的各种查询等。敏感信息可能被完全泄露,也有可能是部分泄露,这取决于敏感信息的安全等级。

推理控制通常是在访问控制的基础上实施的,有效的推理依赖于系统中用户可以合法获取的数据。推理通道(或推理链)是刻画推理问题的一种常用方法,多数推理控制问题可以归结为推理通道的检测和消除。一般来讲,检测并消除推理通道的技术可以分为两类:第一类是在设计阶段发现推理通道,通过修改设计提高某些敏感数据的安全级别来解决,但可能导致给属性或原始数据分配过高的安全级,削弱数据的可用性;第二类是在查询期间消除推理通道,如果检测到潜在的推理通道就拒绝查询或修改查询,以保证敏感信息的安

全。这种基于查询的推理控制问题，代价高，且仍会导致敏感信息的泄漏。

目前常用的推理控制方法可以分为四类：语义数据模型方法、形式化方法、多实例方法和查询限制方法。

1. 语义数据模型方法

语义数据模型方法常用于数据设计中的推理控制。在数据库设计阶段要求设计者确定数据库客体，包括数据、元数据以及约束等及其安全级别。对于数据库设计者和用户来讲，不仅要清楚输入到数据库中数据的安全级别，而且要求在为数据指定安全级别时能够预见到潜在的推理通道。

在理想情况下，为了阻止所有未经授权的信息泄露，应该遵守多级安全数据库的基本规则：一个数据项的安全级别应该支配所有影响它的数据的安全级别。如果一个数据项的值不被其安全级支配的数据影响，信息流就会流向其他安全级。现有的技术主要包括使用安全约束在多安全级数据库设计期间为数据库模式制定适当的安全级，并将安全约束用语义数据模型进行表示。

用于以数据建模技术最早的例子是 Hinke 的 ASD Views 工程。在该工程中，在数据库设计中通过构造语义关系图来表示可能的推理通道。其中，数据项被表示为结点，它们之间的关系由连接结点的边表示。如果两个结点之间存在两条路径，一条路径包含了图中所有的边，而另一条路径不包含图上所有的边，那么两结点之间就有可能存在推理通道。然后进一步分析确认是不是真正的推理通道。相应的解决方法是提升边的级别，直到所有的推理通道都被关闭。这个方法的缺点是提高导致推理问题发生的数据项的安全级别的方法在实际中受到限制。另外，多实例也能提供部分解决方案。

2. 形式化方法

函数依赖定义如下：设 R 是一个关系模式，U 代表关系 R 的属性集，X、Y 是 U 的子集。对于任何关系 R 的实例 r，r 中不存在具有同样的 X 属性值且同时具有不同的 Y 值的两个元组，即如果 $t[X]=s[Y]$，则 $t[Y]=s[Y]$，则称 Y 函数依赖于 X，记作 $X\rightarrow Y$。

如果一个函数依赖对于低安全级别的用户而言是已经知道的，则会产生推理问题。Su 和 Ozsoyoglu 给出了消除函数依赖和多值依赖推理的形式化算法。在该算法中，函数依赖的安全级粒度是属性级，多值依赖的安全级粒度是记录级。对于函数依赖推理，采用提高属性的安全级的算法来消除推理通道；对于多值依赖推理，该算法的核心思想是把存在多值依赖推理的关系实例中的某些元组的安全级升高，经过调整元组的安全级后，新的关系实例不再存在多值依赖推理。

3. 多实例方法

如前所述，多实例允许数据库中存在关键字相同但安全级别不同的元组，即把安全级别作为主关键字的一部分。这样，即使数据库中存在高安全级别的元组，也允许低安全级别的数据插入，从而解决了利用主关键字的完整性进行推理的问题。多实例方法的缺点是使数据库失去了实体完整性，同时增加了数据库中数据关系的复杂性。

4. 查询限制方法

该方法通过分析查询的方法来解决推理通道问题。为了在数据库会话中阻塞推理通道，对用户的查询采取的限制方法主要有修改查询语句和修改查询结果两类。

（1）查询在执行前修改。当系统接收到用户提交的查询时，首先判断该查询是否会导致敏感信息的推理，如果会，那么必须对查询进行转换，使其不能导致敏感信息的导出。Donald G 提出了通过检查用户 SQL 语言来分析推理通道的方法。该方法假定数据库由一个全局关系组成，全局关系可以通过所有关系的笛卡儿积得到。从查询、谓词和模式在表示数据库元组时等价的角度出发，通过检查 SQL 语言的 Where 子句，可以发现大部分推理通道。

（2）修改查询结果。方法（1）虽然阻止了用户进行非法查询，但恶意用户可以通过比较合法查询与非法查询的区别导出推理信息。因此，通过引入不确定性对查询结果进行修改成为研究的重点。图 4-3 是一种典型数据库推理问题控制器原型（DBIC System）的示意图。

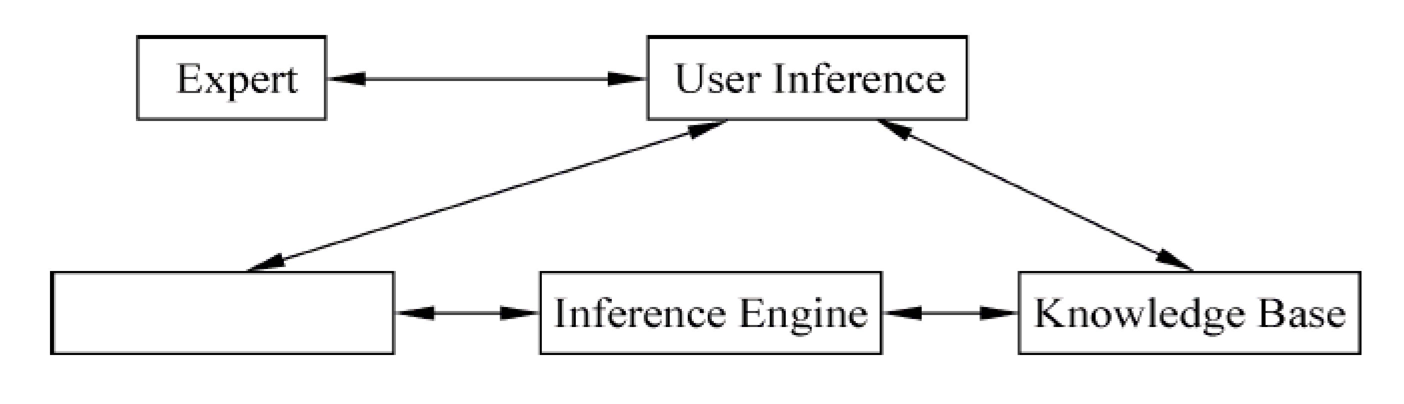


图 4-3　数据库推理问题控制器原型

它是一个由系统安全员使用的基于知识的工具，用来监测和制止推理。推理引擎根据规则进行以下动作：

① 判断哪些查询及相关结果返回给用户；

② 检查返回给用户的数据的完整性和一致性；

③ 启动对推理风险概率的计算并确认推理通道；

④ 根据从用户得来的数据和③的结果修改知识库；

⑤ 对安全标示/安全级别的修改提出建议。

4.6.2　关系数据库的推理控制

判定数据库是否存在推理泄露是对其实施推理控制的前提。利用关联在给定数据集上的推理是一个反复过程，新推理得到的数据可能激活其他关联用于进一步的推理，若所有关联作用在某数据集上都不再推理出新数据，则称此时的推理数据集是推理封闭的。

推理闭包：给定数据库 D 的属性集合 X 和关联集合 F，若存在一个 F 中关联的作用顺序从 X 推理出的 X'，使得 F 中任意的关联作用在 X 上的推理结果都是 X' 的子集，则称 X' 是属性集 X 在关联集合 F 上的推理闭包，用 X_F^+ 表示。

通过推理闭包的计算就可以简单判定数据库对于某用户而言是否存在潜在的推理泄露。若某用户在数据库 D 中的可访问的属性集是推理封闭的，则 D 对该用户是推理安全的，否则出现在相应推理闭包中的敏感属性集即为针对该用户的推理控制对象。

保护敏感数据不被直接访问是安全系统的核心，以访问控制为基础的推理控制是系统的一个有效补充，应该避免因破坏访问控制策略而导致的敏感数据的泄露。在满足访问控制策略的基础上，进行推理控制还应遵循以下基本原则：

（1）信息最大可用原则。实施推理控制后，用户对数据库的访问必将受到新的限制，部分信息因为可能不再能够被访问而丢失，因此，在保证数据库安全的前提下，要尽可能使数据库中的数据最大可用。

(2) 分阶段控制原则。根据敏感数据的定义,以是否存在查询条件可将推理通道分为两类:独立于数据和依赖于数据。前者与具体数据无关,只要通道存在必然导致泄露发生,因此这类通道都应在数据库设计阶段通过安全级或授权调整来消除;对于后者,从通道得到的推理结果不全是敏感的,适合在数据库运行阶段通过监控实现有选择的推理控制。

设计阶段的推理控制独立于具体数据,不用考虑查询的条件。该阶段采用数据隐藏的方法,即在访问控制策略中限制用户对推理通道中某些属性访问,来消除推理通道。属性的隐藏必然导致其对应数据可用性的下降,根据信息最大可用的原则,需要选择对数据可用性影响最小的属性集。

(3) 短通道优先原则。推理过程用到的关联越多,推理通道就越长,推理结果的不确定性通常也就越大。因此,相对于长通道而言,短通道带来的危害更迅速而且更直接,这类通道应优先在设计阶段消除。

4.6.3 XML 数据库的推理控制

XML 已经成为互联网上进行数据传输和交换的标准,但由于互联网的开放性和脆弱性,XML 数据的安全问题一直是研究人员致力解决的问题,目前关于 XML 推理控制的研究才刚刚起步。

关系数据的推理控制可以分别在数据库设计和运行两个阶段实施,而对于 XML 数据而言,由于面向节点的访问控制对系统性能的影响较为严重,其推理控制适合在设计阶段实施。基本策略是计算无推理泄露的安全视图:首先根据访问控制的要求,形成对应于不同用户或用户组的局部文档视图,然后根据源文档中的约束规则去除视图中可能导致敏感数据泄露的节点或子树。此后,用户对源文档的访问将重定向到相应的安全视图上。由于安全视图同时满足访问控制和推理控制的安全要求,因此可以直接返回用户的查询结果,而不必再进行任何控制,从而提高系统的访问效率。实际的处理过程如图 4-4 所示。

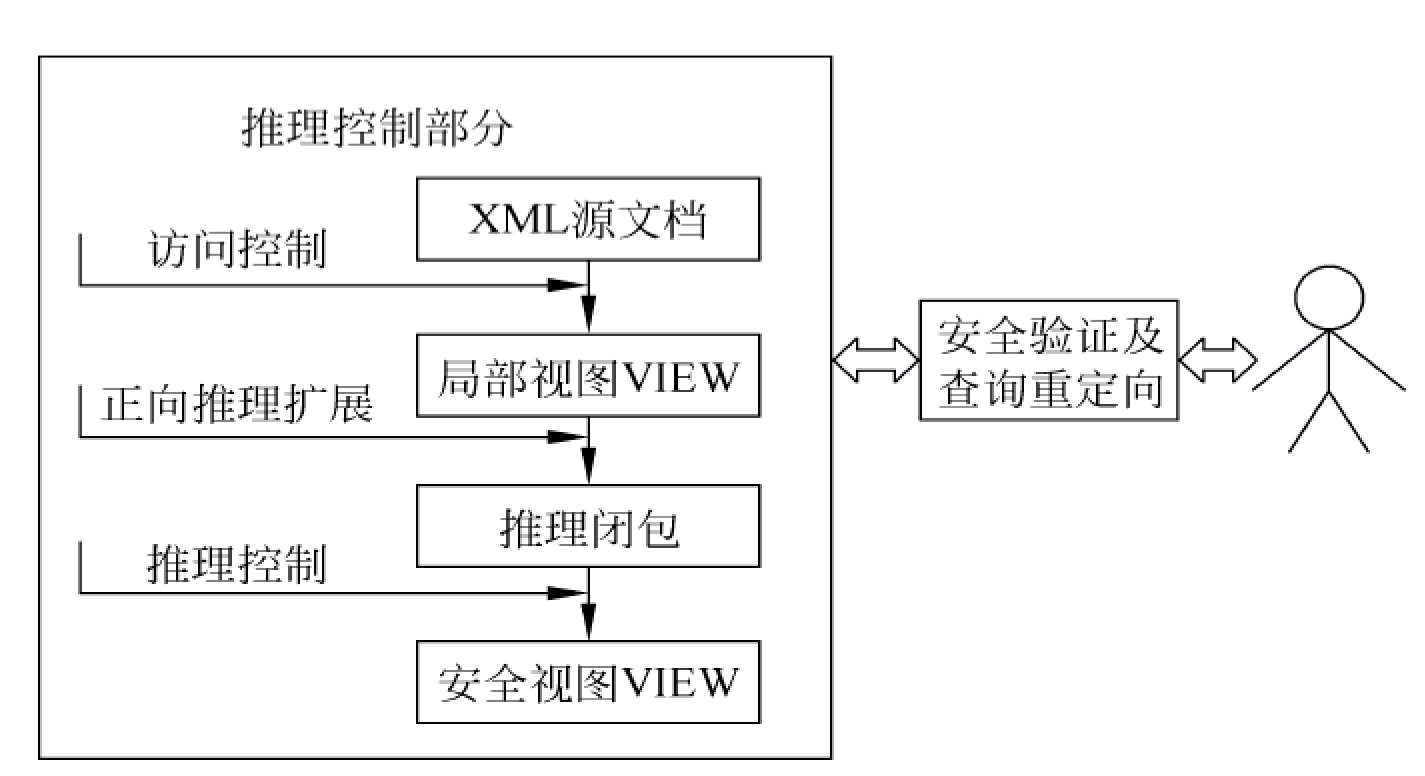


图 4-4　XML 约束推理控制的处理过程示意图

① 指定并保存 XML 源文档中的所有约束规则集合 C。

② 通过访问控制形成对应于不同用户或用户组的源文档局部视图集合 VIEW。

③ 对于 VIEW 中的每一个局部视图 view,利用 C 中的约束对其进行正向推理生成推理闭包 $view^+$。若 $view^+$ 包含敏感数据,则生成敏感数据集 M。

④ 以敏感数据集 M 为基础,进行逆向推理,清除推理链上的所有元素,形成安全无泄

露的用户局部文档视图 view'。

⑤ 用户登录并提交查询,通过验证后将查询映射到相应的局部文档视图 view',由 view'对查询进行响应。

设计阶段的安全视图尽管有着避免复杂授权处理过程以及提高系统运行效率的优点,但同时也存在一定的缺点。首先缺乏灵活性,XML 数据库的任何变动无法直接作用到视图,需要对视图作同步的更新;其次影响系统的可用性,安全视图生成过程中的剪枝处理使得部分有用的数据不再能够被访问;最后对于较大的文档,局部视图的存储和维护也是一个较重的负担。

4.7 隐私保护的数据挖掘

隐私保护技术作为一种新兴的信息安全技术,与传统的访问控制和加密技术有着本质的区别。访问控制技术和加密技术的核心思想是保护数据的隐秘性,保证它不被非授权的第三方访问。一般通过切断从攻击者到隐秘数据的道路(访问控制)或者使得攻击者获得的数据变得不可用(加密技术)来实现。攻击者则以获得可用的隐秘数据为最终目标。而隐私保护技术并不保障资料的隐秘性,隐私资料完全是对外公开的,甚至任何人都可以访问,其核心是要保护隐私数据与个人之间的对应关系,即隐私数据可以被任何人得到,但是却不能把该数据对应到某个特定的人身上。从攻击者的角度看,攻击的目标是隐私数据与个人之间的对应关系。

随着网络、数据库技术的发展,人类大量的个人电子信息被存储、发布,并且用于数据挖掘、知识发现等领域,虽然从海量数据中能够挖掘有用的信息,但是当数据挖掘用于公开分析大量的私人信息(如病史、购物记录、信用记录等)时,它在为人们提供强大知识发现的同时,也给个人隐私带来威胁。

4.7.1 隐私保护数据挖掘的分类

隐私保护数据挖掘是未来数据挖掘一个非常重要的研究领域,从数据分布、数据修改、数据挖掘算法、数据及规则的隐藏以及隐私保护技术五个角度可以对现有的较为典型的隐私保护数据挖掘算法分为如下几类。

(1) 数据分布(Data Distribution):这个角度主要涉及数据的分布方式。

一些算法主要研究数据集中分布的情况,另外一些算法主要研究分布式数据挖掘的情况,其中,分布式数据挖掘又可以分为数据水平分布和数据垂直分布两种。

(2) 数据修改(Data Modification):这个角度主要涉及数据修改的方案。

一般来说,为了确保被公开数据的隐私安全性,原始数据在被公开之前需要经过修改,进行伪装,数据修改方案需要和隐私保护策略相结合。常用的数据修改方法包括以下几种。

① 数据扰乱(Data Perturbation):即将原始数据的属性值替换为一个新的值,如在布尔属性值中进行 0～1 互换,添加噪声数据等。

② 数据阻塞(Blocking):即用“?”来替代存在的值,以保护敏感数据和规则;合并或聚合多个详细数据成为一个更高层次的数据。

③ 取样(Sampling)：即对数据进行抽样。

(3) 数据挖掘算法(Data Mining Algorithm)：这个角度主要考虑适用于不同的数据挖掘算法，如分类算法、关联规则挖掘算法、聚类分析算法等。

(4) 数据及规则的隐藏(Data or Rule Hiding)：这个角度主要考虑需要隐藏的是原始数据还是规则。原始数据主要指用户的姓名、身份证号码、银行卡号码、住址、工资等敏感数据；规则主要指使用数据挖掘算法从数据中挖掘出来的侵犯隐私的信息等。隐私保护数据挖掘的目标是通过使用修正原始数据的算法，使得在进行数据挖掘之后，属于隐私的数据和信息依然保持隐藏而未被发现。对原始数据的保护难点在于，必须在保护隐私的同时，高效地挖掘出数据背后隐藏的知识；而对于规则来说，保护难点在于，隐私保护者希望通过隐私保护算法所隐藏的信息，却经常会被非专业的用户从公开的数据中发现。通常隐藏规则比伪装原始数据的复杂度要高，有时通过保护敏感规则，往往能同时起到保护重要原始数据的目的。

(5) 隐私保护(Privacy Preservation)技术：这个角度主要考虑的是修改数据所采用的技术。隐私保护技术是五个分类标准中最重要的，因为它能直接反映隐私保护的程度。从隐私保护技术这个角度又可以分为：

① 基于启发式技术的方法，例如自适应地进行数据更改技术，该方法只改变部分选中数值，而并非所有数值；

② 基于密码学技术的方法，该方法主要利用密码学知识对数据进行加密，例如安全多方计算(SMC)方法，即参与计算的各方只能获得自己所提供的输入数据以及最终结果，而对于其他参与者的数据一无所知；

③ 基于重构技术的方法，即从被随机扰乱的伪装数据中重构原始数据的分布。

4.7.2　*K*-匿名模型

K-匿名模型要求公布后的数据中存在一定数量的不可区分的个体，使攻击者不能判别出隐私信息所属的具体个体，从而防止了个人隐私的泄密。

设 T 为原始资料表，T' 为要发布的资料表。T 中的属性可以分为以下三类。

(1) 标识符属性：该属性可以唯一的确定一个人，例如身份证号、姓名等。既然此时的目的是为了避免发布过多的信息，以免攻击者可以唯一地识别一个人的信息，因此，这类属性在预处理的时候应该被移出或者加密。

(2) 准标识符属性(QI)：假定一个实体集 U，一个特定的实体表 $T(A_1, A_2, \cdots, A_n)$，$f_c: U \to T$ 以及 $f_g: T \to U'$(其中 $U \in U'$)。T 的一个准标识符 QI 是一组属性 $\{A_i, \cdots, A_j\} \in \{A_1, A_2, \cdots, A_n\}$，其中 $\exists P_i \in U$，那么 $f_g(f_c(P_i)[QT]) = P_i$ 成立。准标识符属性是攻击者已知的，可能同时存在于发布的资料表与其他攻击者可以获得的外部的数据表，攻击者可以利用准标识符属性引起链接攻击，推演出个体信息。准标识符属性不含有任何敏感信息。

(3) 敏感属性：敏感属性是攻击者所不知道的，需要被保护的属性，例如健康状况、个人收入等。

为了保护个人隐私，在数据发布的时候，必须从数据表中删除标识符属性值，但是它依然有可能造成隐私泄漏。例如，在马萨诸塞州，研究者可以从 Group Insurance Commission (GIC)得到人们的健康信息表(MedicalData)，其中包含邮政编码、性别、生日、健康状况等一百多个属性，同时研究者还可以很容易地得到注册选民信息表(VoterList)。这两个表是

公开的，其中虽然健康信息表含有健康状况这个敏感属性，但是该表在发布的时候已经将标识符属性（姓名、身份证号等）删除掉，而注册选民信息表的所有字段属性本身没有任何敏感属性，且是完全公开的。这两个表各自本身没有任何隐私泄露的问题，但是注册选民信息表和健康信息表中含有公共属性字段{Zip，Birthdate，Sex}，如图 4-5 所示，通过对两个表进行链接操作（Select MedicalData. *，VoterList. * from MedicalData As A，VoterList As B where A. Zip＝B. Zip and A. Birthdate＝B. Birthdate and A. Sex＝B. Sex），就可以得到大部分在选民信息表中出现的人的身体健康状况。可见由于数据发布后，数据提供者既不清楚数据接受者所拥有的其他资源，也不能对数据接受者对数据的使用进行控制，因此如果在发布前，不对数据表做特殊的处理，就不能保障数据所有者的隐私信息不被泄漏。

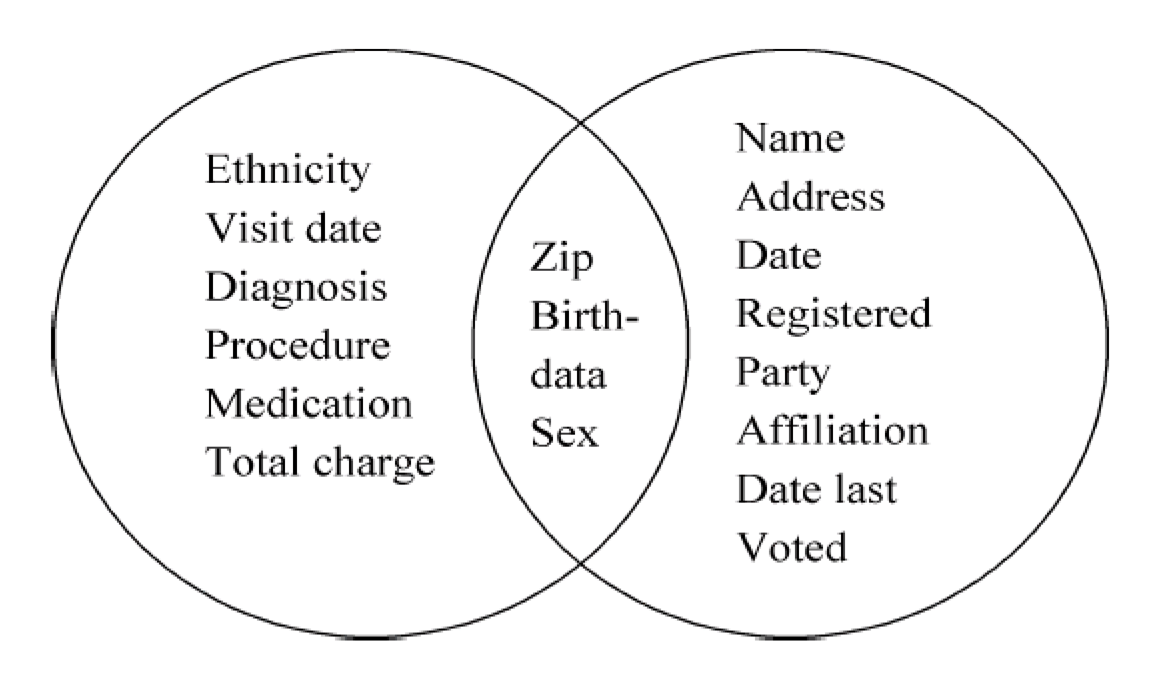


图 4-5　链接攻击

K-匿名：给定数据表 $T(A_1, A_2, \cdots, A_n)$，QI 是与 T 相关联的准标识符，当且仅当在 $T[QI]$ 中出现的每个值序列至少要在 $T[QI]$ 中出现 K 次，则 T 满足 K-匿名。$T[QI]$ 表示 T 表的元组在准标识符 QI 上的投影。

K-匿名化是数据发布环境下保护数据隐私的一种方法。*K*-匿名要求在发布的数据集中，每一条记录 r，都要有至少 K 条记录与其在准标识符属性上具有相同的属性值，即每个等价组所含的元组个数至少为 K。

表 4-4 是一个原始病人基本信息表，表 4-4 的准标识符属性为{Zip code，Age，Nationality}，Condition 为敏感属性。表 4-5 是一个表 4-4 满足 4-匿名化的例子。表 4-5 有 3 个等价组，每个等价组的记录个数至少为 4。*K*-匿名化可以保证攻击者即使知道某一个病人的准标识符属性中有关的信息，但是攻击者能推断出该病人敏感属性具体值的概率不大于 $1/K$。

表 4-4　住院病人信息表

	Non-Sensitive			**Sensitive**
	Zip Code	**Age**	**Nationality**	**Condition**
1	13053	28	Russian	Heart Disease
2	13068	29	American	Heart Disease
3	13068	21	Japanese	Viral Infection
4	13053	23	American	Viral Infection
5	14853	50	Indian	Cancer
6	14853	55	Russian	Heart Disease
7	14850	47	American	Viral Infection
8	14850	49	American	Viral Infection
9	13053	31	American	Cancer
10	13053	37	Indian	Cancer
11	13068	36	Japanese	Cancer
12	13068	35	American	Cancer

表 4-5　满足 4-匿名化的住院病人信息表

	Non-Sensitive			Sensitive
	Zip Code	Age	Nationality	Condition
1	130 **	<30	*	Heart Disease
2	130 **	<30	*	Heart Disease
3	130 **	<30	*	Viral Infection
4	130 **	<30	*	Viral Infection
5	1485 *	≥40	*	Cancer
6	1485 *	≥40	*	Heart Disease
7	1485 *	≥40	*	Viral Infection
8	1485 *	≥40	*	Viral Infection
9	130 **	3*	*	Cancer
10	130 **	3*	*	Cancer
11	130 **	3*	*	Cancer
12	130 **	3*	*	Cancer

如果发布的数据表 T 满足 K-匿名，那么以准标识符 QI 为基础的外部数据表与表 T 的连接，不能够链接到 QI 或 QI 任何一个子集上，来匹配出少于 K 个数目的个体。因此能够预防表中信息与外部数据源的链接推理。

K-匿名模型主要有以下三方面的特点：①匿名策略只针对静态数据，不支持数据更新后的重新发布；②只能提供表级别的保护粒度；③默认所有的属性具有相同的重要性。

原始 K-匿名模型要求一个等价组中至少有 k 条记录，主要是为了防止身份泄露，但是同时会带来属性泄露。Machanavajjhala 等人提出了针对 K-匿名模型的两种攻击方法，一致性攻击和背景知识攻击。一致性攻击是指等价组内的敏感属性值缺乏多样性导致隐私泄露。例如，Alice 和 Bob 是敌对的邻居，有一天 Bob 生病并且被救护车送进医院。Alice 看到后也去医院探个究竟。在医院 Alice 看到公布的一个 4-匿名的现有病人的数据表（见表 4-5），Alice 很快就发现 Bob 所患何病。因为 Alice 是 Bob 的邻居，她知道 Bob 的年龄是 31 岁的美国男子，并且邮编是 13053。因此，Alice 知道 Bob 记录号可能为 9，10，11 或 12。而这几条记录都有一个共同的健康状况值（Cancer），因此 Alice 推断出 Bob 所患为 Cancer。背景知识攻击是指由于攻击者对资料表所知道的附加的背景知识引起的隐私泄露。例如，Alice 有一个笔友 Umeko 和 Bob 住进同一所医院，他的记录同样出现在表 4-4。Alice 知道 Umeko 是一个 21 岁的日本女子，邮编是 13068。通过这些信息，Alice 推出 Umeko 的记录号可能为 1，2，3 或 4，如果没有其他信息的话，Alice 不确定 Umeko 患的是 Virus 或 Heart Disease。但是众所周知日本人患 Heart Disease 的几率非常小，因此 Alice 推断 Umeko 患的应该是 Virus。K-匿名模型在这两种攻击下会造成属性泄露。针对上述两种攻击，Machanavajjhala 给出了通过提高等价组敏感属性多样性的方法（L-diversity）来降低隐私泄露。L-diversity 要求数据表在满足 K-匿名模型的前提下，等价组中至少有 l 条不同的敏感属性值等；由表 4-4 中的住院病人记录表，可以得到一个满足 3-diversity 的数据表（见表 4-6），该数据表可以有效地防止满足 4-匿名化的表 4-5 中遇到的攻击。比如，Alice 不能

从 3-diversity 表中推出 Bob(来自于邮编 13053 的 31 岁的美国男子)患有癌症。尽管 Umeko(一个来自邮编 13068 的 21 岁的日本女子)极不可能患有 Heart Disearse,Alice 仍然不能确定 Umeko 患有 Viral 或 Cancer。

表 4-6 满足 3-diversity 的住院病人信息表

	Non-Sensitive			Sensitive
	Zip Code	Age	Nationality	Condition
1	130 **	≤40	*	Heart Disease
4	1305 *	≤40	*	Viral Infection
9	1305 *	≤40	*	Cancer
10	1305 *	≤40	*	Cancer
5	1485 *	>40	*	Cancer
6	1485 *	>40	*	Heart Disease
7	1485 *	>40	*	Viral Infection
8	1485 *	>40	*	Viral Infection
2	1306 *	≤40	*	Heart Disease
3	1306 *	≤40	*	Viral Infection
11	1306 *	≤40	*	Cancer
12	1306 *	≤40	*	Cancer

4.7.3 匿名化实现技术

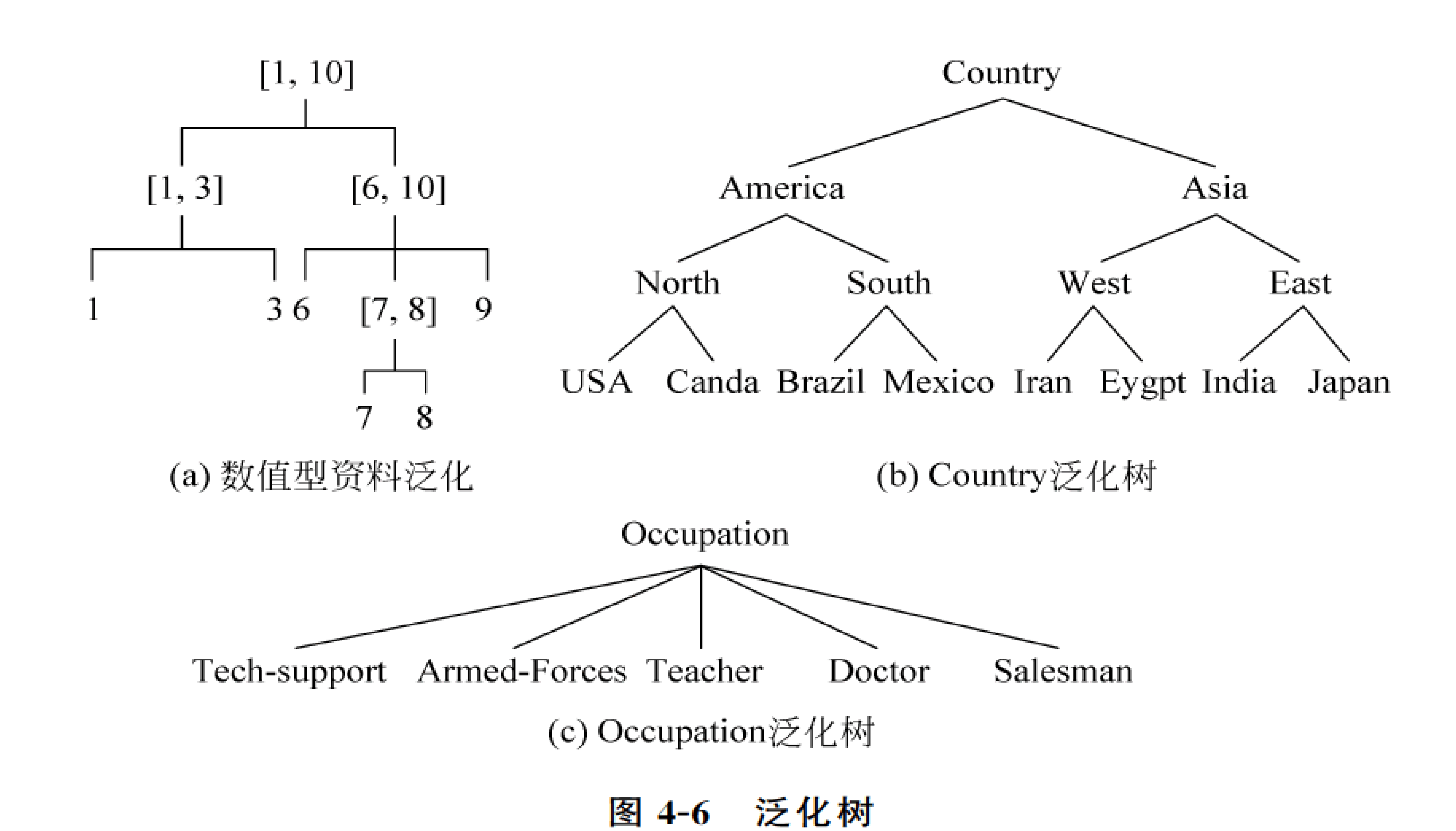
1. 泛化和隐匿

泛化和隐匿是实现匿名化所使用的最早也是最广泛的技术。泛化就是将数据表中的属性的具体值用泛化值来代替,使其意义变得更加广泛。比如将生日从“年/月/日”泛化成“年/月”;将地区从“北京”、“上海”等泛化成“中国”;将数字 12,15,18 泛化成区间[10,20]等。

隐匿,就是直接从数据表中删除一些属性值或记录。隐匿会导致减少了发布的数据,但是在实际应用中可以降低数据的泛化数量,从而降低了数据的信息损失。隐匿通常和泛化结合使用,使没有满足匿名要求的数据记录,通过直接删除来保证数据的安全性。

泛化的过程首先要对每一个属性定义一棵泛化树,泛化树的叶子节点就是具体的属性值,数值型属性泛化树的父节点则是包含所有子节点的一个区间,如图 4-6(a)所示;而对于分类型属性,根据自然地语义关系构造泛化树,其父节点则是含义更为广泛的信息,如图 4-6(b)所示。当属性沿着从叶子节点到根节点的路径进行泛化,越接近根节点,则泛化程度越高,隐私信息的安全性也就越高。但是对于有些属性,如 Occupation,可能不存在任何语义关系来构造泛化树,对于这样的属性,泛化基于一个公共值的属性,可以如图 4-6(c)所示进行泛化。泛化和隐匿的核心思想是以数据的可用性为代价,换取隐私信息的安全性,其过程使得原本不同的 *QI* 属性值变成相同值,从而达到了匿名化的目的,但是它降低了数据的精确性和可用性,减少了记录以及整个数据表所携带的信息量。

通常将实现该算法分成两类:一类是全域泛化,另一类是局域泛化。由于全域泛化研究较早,所以全域泛化比局域泛化要多一些。全域泛化是指同一属性的所有值来自于泛化



(a) 数值型资料泛化　(b) Country泛化树　(c) Occupation泛化树

图 4-6　泛化树

树的同一个层次。例如,生日的所有值要么只有年,要么既有年又有月。全域泛化的一个好处是匿名表有统一的泛化度,但是会造成严重的信息损失。局域泛化是指属性值可以被泛化到泛化树的不同层。可以说局域泛化是一种普通的泛化,全域泛化是局域泛化的特例。

2. 聚类

如果将有 d 个属性的数据表中的 n 条记录看成是 d 维空间中的 n 个点,那么匿名化的求解过程就可以转化成如下聚类问题的求解过程——如何将这 n 个点聚成若干类,并使其满足:①每个类中至少含有 k 个点;②生成类的代价最小。生成的类就相当于等价组,而造成的信息损失量可以用生成类的代价来衡量。

把 K-匿名问题看做聚类问题,在聚类时找到一个聚类集(也就是等价类),每个聚类集中至少包含 K 条记录。为了最大的保证数据质量,需要在每个聚类集中的记录尽可能相似,这样可以保证聚类集在同一聚类中修改为相同准标识符值时,数据损失最小。

K-成员聚类:令 S 是一个有 n 条记录的集合,K 是具体的匿名参数,那么 K 成员聚类问题最佳解决办法是得到一个满足下列条件的聚类集 $\varepsilon=\{e_1,\cdots,e_m\}$:

(1) $\forall\, i\neq j\in\{1,\cdots,m\}, e_i\cap e_j=\Phi$;

(2) $\bigcup_{i=1,\cdots,m}, e_i=S$;

(3) $\forall\, e_i\in\varepsilon, |e_i|\geqslant K$;

(4) $\sum_{l=1,\cdots,m} |e_l|\times \mathrm{MAX}_{i,j=1,\cdots,|e_l|}\Delta(p(l,j),p(l,i))$是最小的。

其中:$|e|$是聚类 e 中资料点的数量;$p(l,i)$表示聚类 e_l 中第 i 个资料;$\Delta(x,y)$是两个数据 x 和 y 的距离。

K-匿名需要找到合适的机制来估算泛化后的信息损失,这种机制不仅能应用于数值型数据,而且能应用于分类型数据的信息损失计算。

数值型数据之间的距离:令 D 是一个有限的数字域,属于 D 两个值 v_i 和 v_j 间的距离正式定义为:$\delta_N(v_1,v_2)=|v_1-v_2|/|D|$。其中$|D|$是属性的域范围,由 D 中最大值、最小值之间的差值来确定。

比如表 4-7 中的记录集。第一个记录和第二个记录中属性 Age 的距离可以这样计算:

$|57-41|/|57-24|=0.485$，而最后两个记录中属性 Age 的距离等于 $|24-45|/|57-24|=0.636$。两个值之间的距离越小意味着彼此越相似。

表 4-7 基本信息表

	Country	Occupation	Age	Salary
R1	USA	Armed-Forces	41	7000
R2	India	Tech-support	57	8000
R3	Canada	Teacher	40	5000
R4	Iran	Tech-support	38	5000
R5	Brazil	Doctor	24	7000
R6	Mexico	Salesman	45	8000

分类型数据之间的距离：令 D 是一个分类域，T_D 是 D 上的一个分类树，$H(T)$ 表示树 T 的高度，$\wedge(V_i,V_j)$ 表示 V_i 和 V_j 在分类树中具有最小相同公共祖先的子树，D 中两个值 v_i 和 v_j 间的距离为

$$\delta_C(v_i,v_j) = H(\wedge(V_i,V_j))/H(T_D)$$

例如，图 4-6(b)中 Country 的泛化树，泛化树中 India 和 USA 的距离是 $3/3=1$，而 India 和 Iran 的距离是 $2/3=0.66$。属性图 4-6(c) Occupation 和它的分类树只有一层，那么两个不同值之间的距离始终是 1。

两个记录间距离：令表 T 的准标识符为 $QI=\{N_1,\cdots,N_m,C_1,\cdots,C_n\}$，$N_i(i=1,\cdots,m)$ 是数值型属性，$C_j(j=1,\cdots,n)$ 是分类型属性，$r_i[A]$ 表示属性 A 在记录 r_i 中的值，δ_N 表示定义 3 中的距离函数，δ_C 表示定义 4 中的距离函数，那么两个记录 r_1 和 r_2 的距离为

$$\Delta(r_1,r_2) = \sum_{i=1,\cdots,m}\delta_{N_i}(r_1,r_2) + \sum_{j=1,\cdots,m}\delta_{C_j}(r_1,r_2)$$

信息损失：令 $e=\{r_1,\cdots,r_k\}$ 是一个聚类（也就是等价类），这里准标识符有数值型属性 $(N_1,\cdots,N_m)$ 和分类属性 $(C_1,\cdots,C_n)$，T_{C_i} 是由分类属性 C_i 的域定义的分类树，MIN_{N_i} 和 MAX_{N_i} 分别是 e 中关于属性 N_i 的最大值和最小值，令 $\cup C_i$ 是 e 中关于属性 C_i 的值的并集，那么泛化 e 产生的信息损失量，可以表示为 $IL(e)$：

$$IL(e) = |e| \times D(e)$$

$$D(e) = \sum_{i=1,\cdots,m}\frac{(\mathrm{MAX}_{N_i}-\mathrm{MIN}_{N_i})}{|N_i|} + \sum_{j=1,\cdots,m}\frac{H(\wedge(\cup C_j))}{H(T_{C_j})}$$

其中：$|e|$ 是 e 中记录的个数；$|N|$ 表示数值域的范围；$\wedge(\cup C_j)$ 是在 $\cup C_j$ 中每个值具有最低的公共祖先的子树；$H(T)$ 是树的高度；MAX_{N_i} 和 MIN_{N_i} 表示 e 中关于属性 N_i 的最大值和最小值。

整体信息损失：令 $\varepsilon=\{e_1,\cdots,e_m\}$ 是匿名表 AT 中所有等价类集合，那么表 AT 的整体信息损失程度的公式为

$$\mathrm{Total}-IL(AT) = \Sigma_{e\in\varepsilon}IL(e)$$

3. 交换

扰乱和添加噪音等技术，都是保护隐私信息的常用技术，在数据挖掘等领域早已有过广泛的应用，这些方法理论上可以保护资料的隐私，但是在实际应用中发布的数据已经失真，失去发布的意义，因此不能直接应用于数据发布。K-匿名模型提出后，交换技术又有了新的突破，凭借其独特的优势成为了面向发布的数据表匿名化的重要方法，并得到了研究者的普遍关注。

泛化和聚类的匿名化技术虽然过程不同，但是它们都使用了泛化的方法，通过将 *QI* 属性值从特殊变得一般，使 *QI* 属性和隐私属性在记录中的一对一关系便成了在等价组中的多对多关系，从而达到匿名化。而基于交换的匿名化技术则不需要进行泛化，Anatomy 和 (*k*，*e*)-Anonymity 是典型的采用交换来实现匿名化的模型。Anatomy 和(*k*，*e*)-Anonymity 都是首先将数据表划分成若干等价组，Anatomy 把 *QI* 属性和隐私属性分成两个表后不进行泛化就直接发布，两者之间通过“组号”来链接，而(*k*，*e*)-Anonymity 则随机调换组内记录之间 *QI* 属性与隐私属性，这样记录的 *QI* 属性将保持原值，但任何一条记录都有一组隐私属性值，并且等价组内隐私属性值的范围至少为 *e*。分别举例作以下说明。

(1) Anatomy 方法。对于表 4-8 的原始资料表，Anatomy 发布一个准标识符表(QIT)和一个敏感属性表(ST)，这两个表将准标识符属性和敏感属性分开。两个表之间通过组编号 groupID 连接。表 4-9(a)和表 4-9(b)分别表示 QIT 和 ST。Anatomy 能够防止隐私泄露是因为 QIT 的每个元组没有包含敏感属性值，只能从 ST 表中猜出敏感属性值的概率。假如攻击者知道 Bob 的年龄 23 岁，zipcode 是 11000，因此从 QIT(见表 4-6(a))中，攻击者得出元组 1 是 Bob，但是并没有得出关于他的任何疾病信息，只得到元组 1 的组编号。从 ST(见表 4-9(b))中，攻击者得出在 groupID 为 1 的 4 个元组中，50% 是 dyspepsia(或 pneumonia)。除此之外，攻击者并没有得到更多的信息。因此攻击者只能推出 Bob 患有 dyspepsia(或 pneumonia)的概率是 50%，无法确定所患何种疾病，在一定程度上，保护了用户的隐私。

表 4-8 原始资料表

age	sex	zipcode	disease
23	M	11000	pneumonia
27	M	13000	dyspepsia
35	M	59000	dyspepsia
59	M	12000	pneumonia
61	F	54000	flu
65	F	25000	gastritis
65	F	25000	flu
70	F	30000	bronchitis

(2) (*k*，*e*)-Anonymity。表 4-10 为原始资料表，表 4-11 为匿名化的符合(3，2000)-Anonymity 的工资表。在这里 $k=3$，指每个等价组至少含有 3 条记录；$e=2000$，指每个等价组 salary 的区间范围至少为 \$2000。通过元组交换，元组所对应的敏感属性值已经发生变化，每个等价组对应一组敏感属性值。假如攻击者知道 Alex 年龄 35 岁，zipcode 为 27101，他可以推断出 Alex 的工资在 \$54000 与 \$56000 之间，但是他无法推断出 Alex 的准确工资。*e* 的值越大，隐私保护程度越高。

Anatomy 和(*k*，*e*)-Anonymity 的共同特点是：首先，按照将数据表中的记录划分若干组；然后，将 *QI* 属性和隐私属性分割开，并进行交换，使一组 *QI* 属性和隐私属性由原来的一对一关系变成一组 *QI* 属性可能对应到等价组中所有隐私属性的一对多关系，从而达到匿名化的目的。基于交换的匿名化技术的优势在于它能精确地保存 *QI* 属性值，大幅度提高数据的可用性，尤其在集合型查询等应用中的可用性。

表 4-9 Anatomy 匿名表

（a）准标识符表(QIT)

row＃	age	sex	zipcode	groupID
1	23	M	11000	1
2	27	M	13000	1
3	35	M	59000	1
4	59	M	12000	1
5	61	F	54000	2
6	65	F	25000	2
7	65	F	25000	2
8	70	F	30000	2

（b）敏感属性表(ST)

groupID	disease	count
1	dyspepsia	2
1	pneumonia	2
2	bronchitis	1
2	flu	2
2	gastritis	1

表 4-10 工资表

tuple ID	ID	Quasi-identifiers			Sensitive
	name	age	zipcode	gender	salary
1	Alex	35	27101	M	$54000
2	Bob	38	27120	M	$55000
3	Carl	40	27130	M	$56000
4	Debra	41	27229	F	$65000
5	Elain	43	27269	F	$75000
6	Frank	47	27243	M	$70000
7	Gary	52	27656	M	$80000
8	Helen	53	27686	F	$75000
9	Jason	58	27635	M	$85000

表 4-11 (3,2000)-Anonymity 工资表

group ID	tuple ID	Quasi-identifiers			Sensitive
		age	zipcode	gender	salary
1	1	35	27101	M	$54000
1	2	38	27120	M	$55000
1	3	40	27130	M	$56000
2	4	41	27229	F	$65000
2	5	43	27269	F	$75000
2	6	47	27243	M	$70000
3	7	52	27656	M	$80000
3	8	53	27686	F	$75000
3	9	58	27635	M	$85000

习　题　4

1. 数据库安全与操作系统安全的主要区别是什么?
2. 数据库安全主要有哪些安全威胁?
3. 什么是数据库的可靠性和完整性?
4. 什么是敏感数据?
5. 多级数据库的安全问题有哪些?
6. 常用的推理控制方法有哪些?
7. 匿名化有哪些实现技术?

第5章 漏洞检测

对于网络信息系统的安全而言,仅具有事后追查或实时报警功能的安全检测装备是不够的,还需要具备系统安全漏洞扫描能力的事先检查型安全工具。系统漏洞检测又称漏洞扫描,即对重要网络信息系统进行检查,发现其中可被攻击者利用的漏洞。

5.1 漏洞概述

5.1.1 漏洞的危害

漏洞也叫脆弱性(vulnerability),是计算机系统在硬件、软件、协议的具体实现或系统安全策略上存在的缺陷或者不足。漏洞一旦被发现,就可以使用这个漏洞获得计算机系统的额外权限,使攻击者能够在未授权的情况下访问或者破坏系统,从而导致危害计算机系统安全。

安全漏洞的大量存在给人们带来了不同程度的危害,它们(病毒、木马、蠕虫和黑客攻击)绝大多数是利用网络系统或软件中存在的安全漏洞来达到破坏系统、窃取机密信息等目的。例如2007年4月出现的ANI.Gen是一个利用微软Windows系统ANI文件处理漏洞(MS07-017)进行传播的网络蠕虫,该漏洞的利用程序通常伪装成一个图片,只要点击了带有恶意代码图片的网站或邮件就会被感染上恶意程序。没有安装补丁的机器一旦打开了包含恶意代码的网站或邮件,病毒或恶意程序就会立即在后台运行,在没有任何反应的情况下使用户的机器中上盗号木马、恶意广告软件、蠕虫病毒等。

5.1.2 漏洞产生的原因

漏洞的成因很多,一般有以下几类。

1. 网络协议漏洞

TCP/IP协议组是目前使用最为广泛的网络互连协议之一。但TCP/IP协议在设计时是将它置于可信的环境之下,并将网络互连和开放性作为首要考虑的问题,而没有过多的考虑安全性。这就造成了TCP/IP协议族本身的不安全性,导致一系列基于TCP/IP的网络服务的安全性也相当脆弱。

2. 应用软件系统漏洞

任何一种软件系统都或多或少存在一定的脆弱性,安全漏洞可以看做是已知的系统脆弱性。例如,一些程序只要接收到一些异常或者超长的数据和参数,就会导致缓冲区溢出。这是因为很多软件在设计时忽略或者很少考虑安全性问题,即使在软件设计中考虑了安全性,也往往因为开发人员缺乏安全培训或没有安全经验而造成了安全漏洞。应用软件系统的漏洞有两种:一是由于操作系统本身设计缺陷带来的安全漏洞,这种漏洞将被运行在该

系统上的应用程序所继承；二是应用软件程序的安全漏洞。

3. 配置不当引起的漏洞

在一些网络系统中忽略了安全策略的制定，即使采取了一定的网络安全措施，但由于系统的安全配置不合理或不完整，安全机制没有发挥作用；或者在网络系统发生变化后，由于没有及时更改系统的安全配置而造成安全漏洞。

5.2　漏洞的分类标准和分级规范

5.2.1　漏洞的分类标准

系统安全漏洞主要概括为以下几方面特征属性：漏洞被攻击者利用的方式，漏洞形成的主要原因，漏洞对系统安全造成的危害，漏洞对系统安全造成的直接威胁，漏洞的触发条件，漏洞的操作角度，漏洞的发生时序。下面根据这几个方面对系统漏洞进行分类。

1）根据漏洞被攻击者利用的方式分类

本地漏洞：攻击者是系统本地的合法用户或已经通过其他攻击方法获得了本地权限的非法用户。攻击者必须在本机拥有访问权限前提下才能发起攻击的漏洞。比较典型的是本地权限提升漏洞，这类漏洞在 UNIX 系统中广泛存在，能让普通用户获得最高管理员权限。

远程漏洞：攻击者是指通过网络，对连接在网络上的任意一台机器进行的攻击。可分为入侵攻击与破坏攻击两种方式。攻击者可以利用并直接通过网络发起攻击的漏洞。这类漏洞危害极大，攻击者能随心所欲地通过此漏洞操作他人的电脑。并且此类漏洞很容易导致蠕虫攻击。

2）根据漏洞形成的主要原因分类

同一系统漏洞，对其不同抽象层次研究，可能会归为不同的形成原因。

输入验证错误(Input Validation Error)：由于未对用户输入数据的合法性进行验证，使攻击者非法进入系统。

缓冲区溢出(Buffer Overflow)：由于向程序的缓冲区中录入的数据超过其规定长度，造成缓冲区溢出，破坏程序正常的堆栈，使程序执行其他命令。

设计错误(Design Error)：由于程序设计错误而导致的漏洞。其实，大多数的漏洞都属于设计错误。

意外情况处置错误(Exceptional Condition Handling Error)：由于程序在实现逻辑中没有考虑到一些意外情况，而导致运行出错。

访问验证错误(Access Validation Error)：由于程序的访问验证部分存在某些逻辑错误，使攻击者可以绕过访问控制进入系统。

配置错误(Configuration Error)：由于系统和应用的配置有误，或配置参数、访问权限、策略安装位置有误。

竞争条件(Race Condition)：由于程序处理文件等实体在时序和同步方面存在问题，存

在一个机会窗口使攻击者能够施以外来的影响。

环境错误(Condition Error):由于一些环境变量的错误或恶意设置造成的漏洞。

3) 根据漏洞对系统安全造成的危害分类

根据漏洞对系统安全造成的危害可分为:获得普通用户权限,获得其他用户权限。

4) 根据漏洞对系统安全造成的直接威胁分类

根据漏洞对系统安全造成的直接威胁可分为:普通用户访问权限,权限提升,本地管理员权限,远程管理员权限,本地拒绝服务,远程拒绝服务,服务器信息泄露,远程非授权文件存取,读取受限文件,口令恢复,欺骗等。

事实上一个系统漏洞对安全造成的威胁远不限于它的直接可能性,如攻击者获得了系统的普通用户访问权限,就极有可能利用本地漏洞升级为管理员权限。

5) 根据漏洞的触发条件分类

主动触发漏洞,攻击者可以主动利用该漏洞进行攻击,如直接访问他人计算机。

被动触发漏洞,必须要计算机的操作人员配合才能进行攻击利用的漏洞。比如攻击者给管理员发一封邮件,带了一个特殊的 jpg 图片文件,如果管理员打开图片文件就会导致看图软件的某个漏洞被触发,从而系统被攻击,但如果管理员不看这个图片则不会受攻击。

6) 根据漏洞的操作角度分类

文件操作类型,主要为操作的目标文件路径可被控制(如通过参数、配置文件、环境变量、符号链接等),这样就可能导致下面两个问题:写入内容可被控制,从而可伪造文件内容,导致权限提升或直接修改重要数据(如修改存贷数据),内容信息可被输出,包含内容被打印到屏幕、记录到可读的日志文件、产生可被用户读的 core 文件等。

内存覆盖,主要为内存单元可指定,写入内容可指定,这样就能执行攻击者想执行的代码(缓冲区溢出、格式串漏洞、PTrace 漏洞、历史上 Windows 2000 的硬件调试寄存器用户可写漏洞)或直接修改内存中的机密数据。

逻辑错误,这类漏洞广泛存在,但很少有范式,所以难以察觉,可细分为:外部命令执行问题,典型的有外部命令可被控制(通过 PATH 变量,输入中的 SHELL 特殊字符等)和 SQL 注入问题。

7) 根据漏洞发生的时序分类

已发现很久的漏洞:厂商已经发布补丁或修补方法,很多人都已经知道。这类漏洞通常很多人已经进行了修补,宏观上看危害比较小。

刚发现的漏洞:厂商刚发补丁或修补方法,知道的人还不多。相对于上一种漏洞其危害性较大,如果此时出现了蠕虫或傻瓜化的利用程序,将会导致大批系统受到攻击。

Oday:还没有公开的漏洞,在私下交易中的。这类漏洞通常对大众不会有什么影响,但会导致攻击者瞄准的目标受到精确攻击,危害也是非常之大。

5.2.2 漏洞的分级规范

系统安全漏洞分类与分级是对漏洞不同抽象层次的特征属性进行描述。某些系统的分类是在系统安全漏洞的分类基础上进行的,如攻击系统要依据系统安全漏洞分类基础进行

分类,还有一些安全事件的处理方法也要依据系统安全漏洞来制定。所以,一套完善的系统安全漏洞分级分类标准是基础。依照前人理论研究的基础,以及对国外权威网站的漏洞信息进行分析整理统计,提出了下面的分级分类标准。

系统安全漏洞根据其对系统造成的潜在威胁(破坏性、危害性、严重性)以及被利用的可能性进行分级,分为紧急、重要、中等、低等四级。

1) 第一等级:紧急

定义:对"紧急"级别漏洞的利用可以导致网络蠕虫和病毒在用户不知情的情况下在网络上任意传播和繁殖。

对于被评为"紧急"的安全漏洞,需要立即安装升级包(补丁程序)。

2) 第二等级:重要

定义:对"重要"级别漏洞的利用可以导致严重的后果,以致危害到用户数据、相关资源的机密性、完整性和有效性。

对于被评为"重要"的安全漏洞,需要安装升级包(补丁程序)。

3) 第三等级:中等

定义:由于默认配置、审核或难以利用等因素的影响,"中等"级别漏洞的利用效果显著降低。

关于"中等"系统安全漏洞则应该在阅读安全信息以后判断该安全漏洞是否对此系统产生影响。在此基础上,确认升级包不会影响到系统之后,可以采用并安装。

4) 第四等级:低等

定义:对于"低等"级别漏洞的利用效果已降至最低限度,漏洞利用难度非常大。

关于"低等"系统安全漏洞则应该在阅读安全信息以后判断该安全漏洞是否对此系统产生影响。在此基础上,确认升级包不会影响到系统之后,可以采用并安装。

几乎所有的用户都不会受到影响的安全性漏洞将被评级为"低等"。但这并不意味着完全不会受到影响,因此需要与"中等"一样,在阅读安全信息以后,根据需要来决定是否采用该升级包。

5.3 漏　洞　库

5.3.1 漏洞库概述

漏洞数据库就是把所有的系统安全漏洞机器相关信息都输到数据库中,方便用户检索自己的系统会有哪些漏洞、可能的危害、如何安装补丁、补丁信息的链接地址等。漏洞数据库的基本功能主要包括以下三点。

(1) 帮助用户确认自身应用环境中可能存在的安全漏洞,提供基本的漏洞信息查询。每当用户引入新的应用并享用其带来的便利时,这个新的应用也可能带来了已知或未知的安全漏洞,给攻击者提供了新的攻击渠道。通过查询一个信息完备的漏洞数据库,用户就可

以方便地了解到自己所使用的应用的已知安全漏洞和解决方法，从而在入侵发生之前修补漏洞，切断攻击通路，挫败可能的攻击。

(2) 安全产品厂商可以基于持续更新的漏洞数据库和用户的应用布置情况，在用户所使用的应用出现新漏洞时，为用户提供及时的安全预警，通知用户采取修补措施，先于攻击者实施入侵堵截，截断攻击者的非法访问途径，避免损失。

(3) 为安全产品厂商基于安全漏洞发现和攻击保护类产品开发提供技术和数据支持。

漏洞数据库的创建和维护是一项需要大量精力的工作。一个好的漏洞数据库应该体现在以下几个方面。

(1) 覆盖面广。网络已经介入到社会生活的方方面面，应用数量众多，而且差异大，好的漏洞数据库应该覆盖尽可能多的应用程序、操作系统和网络设备，使用户能尽可能多地了解攻击者潜在的入侵途径。

(2) 准确性高。用户需要准确的漏洞描述信息，以帮助他们应对因漏洞引起的安全问题。如提供漏洞影响的软件版本信息、漏洞可能造成的威胁情况、是否可远程利用以及相应修补方案，不仅提供补丁下载地址，最好还能提供暂时不能安装补丁时的临时解决方法和相应的操作指示等。

(3) 及时性。漏洞数据库需要最及时地提供信息。每个工作日都应该向用户提供最新的漏洞信息，在攻击者采取行动之前，第一时间帮助用户堵住安全威胁渠道。

(4) 完全性。漏洞数据库应该提供完全的漏洞相关信息，包括提供与漏洞相关的利用方法和攻击代码等。隐藏漏洞相关的攻击方法，无助于为用户提供更多的保护，只会使用户产生漏洞还没有确切利用方法的错觉，从而降低他们对漏洞进行处理的优先级。

漏洞数据库一般包括以下详细信息。

(1) 漏洞基本信息：漏洞名称、CVE 编号、本地编号、发布/更新时间等。

(2) 漏洞分类分级信息：漏洞级别/类型、攻击类型/效果等。

(3) 参考资源信息：权威网站中与此漏洞相关的信息。

(4) 受影响的系统：此漏洞对哪些系统构成威胁的信息。

(5) 关键字：描述此漏洞最关键的信息。

(6) 补丁信息：漏洞补救方法、补丁信息和补丁下载地址信息等。

5.3.2 CVE 漏洞库

大部分安全工具都设置了漏洞数据库，而它们之间存在明显不同，这样当不同数据库指向同一问题时就难以统一，造成安全覆盖面存在潜在的差异，而且不同数据库和工具之间互用性差。国际漏洞标准(Common Vulnerabilities and Exposure，CVE)的出台解决了这一问题，使不同的漏洞数据库和安全工具之间彼此交流，更容易共享数据库，方便在其他数据库中搜索信息。通过此标准可以评估每个工具的安全覆盖程度，从而可以判定安全工具的有效性和适应性。CVE 是安全漏洞及其他信息安全风险标准名称的列表，目的是使所有已知漏洞和安全风险的名称标准化。

CVE 的目标是对所有的公认的漏洞和安全暴露建立一个命名标准列表，它使用了通用

的命名，使得所有符合 CVE 标准的安全产品能够通过数据库和工具共享数据。CVE 的建立有以下的特点：①一个漏洞或者一个暴露有自己单独的名称，如上面提到的 CVE-1999-0014；②对于目前出现的每个漏洞和暴露都进行了标准描述；③它是一个字典，而不是数据库；④能够让分开的数据库和安全工具用同一种语言"交谈"；⑤提供了一种协同工作的方式，安全覆盖面更广；⑥提供了安全工具和数据库安全评估的标准；⑦方便从 Internet 上下载和讨论；⑧通过 CVE 编辑板得到行业认可。

CVE 对所有的漏洞和暴露的描述都是通过条目（item）来命名的。CVE 所有的条目中存在两种状态，一种就是正式（entry）条目，另外一种是候选（candidate）条目。当一个漏洞被发现后（通过全球用户的反映，这个时候还不能说是漏洞，只能说是安全威胁），这个威胁就给冠以候选条目的状态，写入 CVE 的条目列表。CVE 编辑板的各个专家就可以对新加入的候选条目进行评价和投票来决定这个条目是否可以成为正式的 CVE 条目。如果能够成为正式条目，则发布到 CVE 的网站上。

每个漏洞都有自己的 CVE 条目与之对应，所以在漏洞数据库中必须存放与之对应的 CVE 条目信息；创建该漏洞库的目的不仅仅是展示条目给用户，而是帮助用户解决漏洞，因此，可以在漏洞数据库中加入解决方法字段；另外可以加入检测漏洞的程序名等信息，让用户能够采用其余的工具来检测是否具有这个漏洞。表 5-1 给出了一个漏洞数据库表应有的结构。

表 5-1　漏洞数据库的表结构

字　段　名	数 据 类 型	说　　明
CVEID(PK)	char	CVE 标准中对这个漏洞的编号
Name	char	漏洞的名称
Eigenvalue	char	漏洞的特征值
TestMethod	char	测试方式或者测试工具名称
CVEDes	char	CVE 中对这个漏洞的描述
Repair	char	建议修补方案以及补丁地址
Port	char	受影响的端口号

5.3.3　其他漏洞库

其他比较著名的网络安全公司或组织，如 Security focus 公司、ISS 公司、AUSCERT 和 DEBIAN 等也都建立了自己的漏洞数据库，并提供了相对完备的漏洞信息和多样的查询手段。

我国在这方面的研究还处于起步阶段，有关漏洞的知识和整理分析还不完备，必须尽快建立符合我国实际、权威、完备的漏洞库，以满足我国信息产业的发展需要。

下面给出的是一些权威的漏洞库站点：①http://www.securityfocus.com 国外著名漏洞发布站点，即 Bugtraq；②http://www.cert.org 美国计算机应急响应小组；③http://www.nipc.org.cn 国家计算机网络入侵防范中心；④http://nsfocus.com 绿盟科技；⑤http://www.xfocus.net 网络安全焦点。

5.4 扫 描 器

5.4.1 扫描器的基本概念

扫描器是一种自动检测远程或本地系统安全性弱点(漏洞)的程序。目标可以是工作站、服务器、交换机、数据库应用等各种对象。例如通过扫描器可以发现远程服务器的各种TCP端口的分配及提供的服务和它们的软件版本,这就间接或直接地了解到远程主机所存在的安全问题。对于管理者来说,扫描器是一种网络安全性评估软件,安全扫描技术与防火墙、安全监控系统互相配合能够提供安全性很高的网络;对于攻击者来说,扫描器是重要的搜集信息的工具。

安全扫描可以根据完整的安全漏洞集合,进行全面的检测,而这些安全漏洞集合也正是导致网络遭受破坏的主要因素,因此,网络安全扫描可以在网络漏洞被利用之前协助管理者及早发现。

根据不同的标准,安全扫描器主要可以分为以下两类。

1) 主机安全扫描器和网络安全扫描器

这种分类方法是从扫描器整体结构和采用的扫描检测方法的不同出发的。主机型安全扫描器的代表有COPCS、Tripwire、Tiger和ISS SystemScanner等。网络型安全扫描器的代表有ISS、Satan和Nessus等。

2) 端口安全扫描器和漏洞安全扫描器

这种方法是以扫描器检测对象的性质不同作为出发点的。端口安全扫描器检测目标主机的任意TCP/UDP端口,来查看目标主机提供了哪些服务(即哪些服务是激活的),另外,一般端口扫描器还包含主机状态扫描功能,即具备解析扫描和Ping扫描功能。漏洞安全扫描器自动检测目标主机和网络上设备的安全漏洞,并且大部分漏洞安全扫描器都具备端口安全扫描器的部分或者全部功能。

5.4.2 主机扫描

主机扫描的目的是确定在目标网络上的主机是否可达。

常用的传统扫描手段有如下几种。

(1) ICMP Echo扫描。发送ICMP Echo请求,然后等待ICMP Echo应答。如果收到了应答,就认为目标是激活状态。其实就是使用常规的ping命令,ping在判断一个网络上主机是否开机时非常有用。向目标主机发送ICMP Echo Request (type 8)数据包,等待回复的ICMP Echo Reply包(type 0)。如果能收到,则表明目标系统可达,否则表明系统已经不可达或发送的包被对方的设备过滤掉。如果想阻止对这样的ICMP Echo应答,只需要禁止ICMP Echo即可。

如果需要扫描一个主机上甚至整个子网上的成千上万个端口,首先判断一个主机是否开机非常重要,这就是Ping扫描器的目的。

(2) ICMP Sweep扫描。可以通过并行发送,同时扫描多个目标主机,以提高扫描效

率，这叫 ICMP Sweep 扫描。

(3) Broadcast ICMP 扫描。向整个局域网发送 ICMP Echo 请求。这样的请求会被广播到整个局域网，网中激活的主机会送 ICMP Echo 应答。UNIX 系统对于请求常常会送网络地址，而 Windows 系统常常忽略请求。

(4) Non-Echo ICMP 扫描。一些其他 ICMP 类型包也可以用于对主机或网络设备的扫描，如 Stamp Request(Type 13)、Reply(Type 14)、Information Request(Type 15)、Reply(Type 16)、Address Mask Request(Type 17)、Reply(Type 18)。

防火墙和网络过滤设备常常导致传统的扫描手段变得无效。为了突破这种限制，必须采用一些非常规的手段，利用 ICMP 协议提供网络间传送错误信息的手段，往往可以更有效地达到目的。非常规的扫描手段有以下几种。

(1) 异常的 IP 包头。向目标主机发送包头错误的 IP 包，目标主机或过滤设备会反馈"ICMP Parameter Problem Error"消息。常见的伪造错误字段为 Header Length Field 和 IP Options Field。根据 RFC1122 的规定，主机应该检测 IP 包的 Version Number、Checksum 字段，路由器应该检测 IP 包的 Checksum 字段。不同厂家的路由器和操作系统对这些错误的处理方式不同，返回的结果也各异。如果结合其他手段，可以初步判断目标系统所在网络过滤设备的访问列表 ACL。

(2) 在 IP 头中设置无效的字段值。向目标主机发送的 IP 包中填充错误的字段值，目标主机或过滤设备会反馈"ICMP Destination Unreachable"消息。这种方法同样可以扫描目标主机和网络设备以及其 ACL。

(3) 错误的数据分片。当目标主机接受到错误的数据分片(如某些分片丢失)，并且在规定的时间间隔内得不到更正时，将丢弃这些错误数据包，并向发送主机反馈"ICMP Fragment Reassembly Time Exceeded"错误消息。利用这种方法同样可以检测到目标主机和网络过滤设备及其 ACL。

(4) 通过超长包扫描内部路由器。若构造的数据包长度超过目标系统所在路由器的 PMTU 且设置禁止分片标志，该路由器会反馈"Fragmentation Needed and Don't Fragment Bit was Set"错误消息，从而获取目标系统的网络拓扑结构。

(5) 反向映射扫描。该技术用于扫描被过滤设备或防火墙保护的网络和主机。通常这些系统无法从外部直接到达。如想扫描某个未知网络内部的机构，可以构造可能的内部 IP 地址列表，并向这些地址发送数据包。当对方路由器接收到这些和数据包时，会进行 IP 识别并路由，对不在其服务范围的 IP 包发送"ICMP Host Unreachable"或"ICMP Time Exceeded"错误消息，没有接收到相应错误消息的 IP 地址会被认为在该网络中。当然，这种方法也会受到过滤设备的影响。

5.4.3 漏洞扫描

漏洞扫描是指检测远程或本地系统存在的安全缺陷。网络安全漏洞扫描器是指在 Internet 上通过向远程或本地主机发送探测数据包，获取主机的响应，并根据反馈的数据包，进行解包和分析，从而检测目标网络或主机系统漏洞的程序。简单地说，扫描器的工作原理就是模拟攻击者的手法主动地探测目标系统，发现和分析网络系统中可能存在的各种安全隐患，将扫描结果报告给用户，并向用户提供该漏洞的相应解决方法，从而保障网络和

系统的安全，避免网络遭受恶意用户再次利用该漏洞实施的攻击。

一个漏洞扫描器应该有如下几个主要的功能：①探测扫描目标主机并且对其工作状态进行识别；②对于正在运行的主机，能够识别开放的端口的状态及提供的服务；③识别目标主机系统及服务程序的类型和版本，获取相应主机的系统信息；④根据已知漏洞信息，进行脆弱性分析。

安全漏洞扫描通常采用两种策略，第一种是被动式策略，第二种是主动式策略。所谓被动式策略就是扫描基于主机之上，从一个内部用户的角度来检测操作系统级的漏洞，主要用于检测注册表和用户配置中的漏洞，基于主机的扫描优势在于它能直接获取主机操作系统的底层细节，如特殊服务和配置的细节等，其缺点在于只有控制了目标主机，并将检测工具安装在目标主机，才能实施正常的扫描活动；而主动式策略是基于网络的，它通过执行一些脚本文件、模拟对系统进行攻击的行为并记录系统的反应，从而发现其中的安全漏洞，它从外部攻击者的角度对网络及系统架构进行扫描，主要用于查找目标开放的端口、提供的网络服务及协议中的漏洞，同时能够有效地发现那些基于主机的扫描所不能发现的网络设备漏洞，如路由器、交换机、远程访问服务和防火墙等存在的漏洞。利用被动式的策略扫描称为系统（主机）安全扫描，利用主动式的策略扫描称为网络安全扫描。

5.4.4 端口扫描

常见的端口扫描方式主要有以下几种。

1. TCP Connect()扫描

这是最简单的扫描方式。扫描程序依次尝试和要扫描的各端口建立正常的 TCP 连接，如果成功，则说明该端口开放，否则端口不开放。

TCP Connect 扫描也称 TCP 全连接扫描，是最基本的 TCP 扫描方法。其原理是，扫描主机通过与扫描的目标机器的制定端口建立一次完整的 TCP 连接，以确定目标主机所开放的端口。扫描主机调用系统的 Connect()函数，尝试与目标主机建立一个 TCP 连接，如果被扫描的端口开放，则连接建立成功；否则，返回－1，表明该端口关闭。通过前面的 TCP 连接建立的三次握手过程可知，若建立 TCP 连接成功，目标主机会发送一个确认报文给扫描主机，其中 SYN＝1，ACK＝1。则扫描主机接收到该报文后，可知目标主机的目标端口处于监听（开放）状态；建立连接失败时，目标主机会向扫描主机发送 RST 响应，表明目标端口处于关闭状态。

TCP Connect()扫描的优点是不需要任何特殊的权限，系统任何用户都可以调用 Connect()函数；且该方法的速度快。但 TCP Connect()方法的缺点是它很容易被发觉，并且容易被防火墙过滤掉，并且目标主机的日志文件会记录一系列有关该服务的连接信息。

利用 nmap 的-sT 选项可以进行 TCP Connect()扫描。对 202.119.201.74 进行扫描，结果如图 5-1 所示。

通过扫描结果可以得到 202.119.201.74 的 4 个端口的信息。

2. TCP SYN 扫描

这种技术通常认为是“半开放”扫描，因为它不完成一次完整的 TCP 连接。首先扫描程序发送一个 SYN 数据包，假装要打开一个实际的连接并等待反应。如果返回 Synlack 信息

```
C:\>nmap -sT 202.119.201.74

Starting Nmap 4.68 ( http://nmap.org ) at 2008-07-16 10:56 中国标准时间
Interesting ports on 933aa7424702401 (202.119.201.74):
Not shown: 1711 filtered ports
PORT     STATE SERVICE
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
1025/tcp open  NFS-or-IIS
MAC Address: 00:0C:29:6A:B5:0B (VMware)

Nmap done: 1 IP address (1 host up) scanned in 76.547 seconds
```

图 5-1　nmap TCP Connect( )扫描

就表示端口处于侦听状态。如果返回 RST 则表示端口没有处于侦听态。假如收到一个 Synlack，则扫描程序就发送一个 RST 信号来关闭这个连接过程而不是发送 ACK 包。这种扫描技术的优点在于一般不会在目标计算机上留下记录。但这种方法的一个缺点是，必须要有管理员权限才能建立自己的 SYN 数据包。

TCP SYN 扫描通常称为“半开放扫描”，因为扫描程序不需要建立一个完整的 TCP 连接。扫描主机向选定的目标主机端口发送一个(“我可以连接到你吗？”)SYN 数据包，好像准备打开一个实际的连接并等待回应一样。若目标主机返回 SYN/ACK 数据包则表示目标端口处于监听状态；而若返回 RST 数据包则表示该端口没有开放。如果收到的是一个 SYN|ACK 数据包，则扫描程序必须再发送一个 RST 信号给目标主机，终止建立连接。因此在 SYN 扫描中并未建立全连接。

TCP SYN 扫描的优点是一般不会在目标主机上留下记录，但需要有管理员权限才能建立自己的 SYN 数据包。

利用 nmap 的-sS 选项可进行 TCP SYN 扫描。对 202.119.201.74 进行扫描，结果如图 5-2 所示。

```
C:\>nmap -sS 202.119.201.74

Starting Nmap 4.68 ( http://nmap.org ) at 2008-07-16 16:23 中国标准时间
Interesting ports on cumt-bc035edd4d (202.119.201.74):
Not shown: 1711 closed ports
PORT     STATE SERVICE
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
1025/tcp open  NFS-or-IIS
MAC Address: 00:0C:29:6A:B5:0B (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.594 seconds
```

图 5-2　TCP SYN 扫描

3. TCP FIN 扫描

有时使用 SYN 扫描都不能保证扫描行为不被发现。一些防火墙和包过滤器会对一些指定的端口进行监视，有的程序能检测到这些扫描。而 FIN 数据包可能会没有任何麻烦地通过。这种扫描方法的思想是关闭的端口会用适当的 RST 来回复 FIN 数据包。另一方面，打开的端口会忽略对 FIN 数据包的回复。这种方法和系统的实现有一定的关系。有的系统不管端口是否打开，都回复 RST，这样这种扫描方法就不适用了。并且这种方法在区分 UNIX 和 NT 时十分有用。

TCP FIN 扫描技术不包含标准的 TCP 三次握手协议的任何一部分，比 TCP SYN 扫描更隐蔽，很难记录踪迹。TCP FIN 扫描通过向目标主机发送包含 FIN 标志的 TCP 包探测端口，如果被探测的目标主机的端口处于关闭状态，则目标主机返回一个 RST 信号；如果目标主机的端口处于侦听状态，则目标主机会忽略对 FIN 数据包的回复，即主机不会返回 RST 信号。通过是否返回 RST 信号，可以判定目标主机开放了哪些端口。

利用 nmap 的-sF 选项可进行 TCP FIN 扫描。对 202.119.201.86(Red Hat Linux 9)进行扫描，结果如图 5-3 所示。

```
C:\>nmap -sF 202.119.201.76

Starting Nmap 4.68 ( http://nmap.org ) at 2008-07-19 10:48 中国标准时间
Interesting ports on jhf-cumt (202.119.201.76):
Not shown: 1713 closed ports
PORT     STATE         SERVICE
22/tcp   open|filtered ssh
111/tcp  open|filtered rpcbind
MAC Address: 00:0C:29:B5:56:B9 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 1.797 seconds
```

图 5-3　利用 nmap 的-sF 选项可进行 TCP FIN 扫描

4. TCP Null 扫描

TCP Null 扫描和 TCP FIN 扫描的原理相似。TCP Null 扫描将发送的探测包的所有标志位置 0，如果被探测的目标主机的端口处于关闭状态，则目标主机返回一个 RST 信号；如果目标主机的端口处于侦听状态，则主机不会返回 RST 信号。通过是否返回 RST 信号，可以判定目标主机开放了哪些端口。

利用 nmap 的-sN 选项可进行 TCP Null 扫描。对 202.119.201.86 进行扫描，结果如图 5-4 所示。

```
C:\>nmap -sN 202.119.201.76

Starting Nmap 4.68 ( http://nmap.org ) at 2008-07-19 10:49 中国标准时间
Interesting ports on d5a3f86f26764b8 (202.119.201.76):
Not shown: 1713 closed ports
PORT     STATE         SERVICE
22/tcp   open|filtered ssh
111/tcp  open|filtered rpcbind
MAC Address: 00:0C:29:B5:56:B9 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 1.860 seconds
```

图 5-4　利用 nmap 的-sN 选项可进行 TCP Null 扫描

5. TCP Xmas Tree 扫描

TCP Xmas Tree 扫描只是将探测包标志位中的 FIN、PSH 与 URG 设置为 1。同样，对于 TCP Xmas Tree 的探测报文，如果端口开放则目标主机不返回 RST 信号，若端口关闭则返回 RST 信号。

利用 nmap 的-sX 选项可进行 TCP Xmas Tree 扫描。对 202.119.201.86 进行扫描，结果如图 5-5 所示。

6. TCP Maimon 扫描

sM Maimon 扫描(TCP Maimon 扫描)是用它的发现者 Uriel Maimon 命名的。这项技

```
C:\>nmap -sX 202.119.201.76

Starting Nmap 4.68 ( http://nmap.org ) at 2008-07-19 10:50 中国标准时间
Interesting ports on c8c001b7bead4f9 (202.119.201.76):
Not shown: 1713 closed ports
PORT     STATE         SERVICE
22/tcp   open|filtered ssh
111/tcp  open|filtered rpcbind
MAC Address: 00:0C:29:B5:56:B9 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 1.765 seconds
```

图 5-5　利用 nmap 的-sX 选项可进行 TCP Xmas Tree 扫描

术和 Null、FIN 以及 Xmas 扫描完全一样，除了探测报文中设置的标志位是 FIN 与 ACK。根据 RFC 793(TCP)，无论端口开放或者关闭，都应该对这样的探测响应 RST 报文。然而，Uriel 注意到如果端口开放，许多基于 BSD 的系统只是丢弃该探测报文。

利用 nmap 的-sM 选项可进行 TCP Xmas Tree 扫描。对 202.119.201.86 进行扫描，结果如图 5-6 所示，结果显示未扫出端口。

```
C:\>nmap -sM 202.119.201.76

Starting Nmap 4.68 ( http://nmap.org ) at 2008-07-19 10:55 中国标准时间
All 1715 scanned ports on jhf-cumt (202.119.201.76) are closed
MAC Address: 00:0C:29:B5:56:B9 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.500 seconds
```

图 5-6　利用 nmap 的-sM 选项可进行 TCP Xmas Tree 扫描

7. TCP Fragmentation

前几种扫描都不能通过防火墙，因为防火墙通常只允许以少数几个端口为目的端口的 TCP 报通过，这样就无法达到一次扫描多台主机的目的，但通过把一个 TCP 报分割到多个 IP 包中，可使防火墙无法从一个 IP 包中找到完整的 TCP 报头，从而无法进行过滤。

8. UDP 扫描

UDP 是无连接的协议，不需要建立连接过程，但这个协议不能保证数据的安全到达。

UDP ICMP 端口不能到达扫描：这种方法与上面几种方法的不同之处在于使用的是 UDP 协议。由于这个协议很简单，所以扫描变得相对比较困难。这是由于打开的端口对扫描探测并不发送确认信息，关闭的端口也并不需要发送一个错误数据包。幸运的是，许多主机在用户向一个未打开的 UDP 端口发送数据包时，会返回一个 ICMP_PORT_UNREACH 的错误。这样用户就能发现那个端口是关闭的了。UDP 和 ICMP 错误都不保证能到达，因此这种扫描方法还必须实现一个包，这种包看上去是丢失的时候能重新传输。这种扫描方法速度很慢，因为 RFC 对 ICMP 错误消息的产生速率做了规定。同样，这种扫描方法需要具有 root 权限。

5.5　扫 描 工 具

现在大量的扫描器集成了主机扫描、端口扫描和漏洞扫描的功能，而且还具有攻击功能，常用的有 X-Scan、Fluxay 等。

5.5.1 X-Scan

X-Scan是国内著名网络安全组织“安全焦点”开发的综合扫描工具，能提供漏洞扫描、漏洞溢出程序报告，方便管理和测试网络，下面以X-Scan v3.3演示这款工具的使用。

图5-7是流光v3.3，首先对其参数进行设置，这里设置扫描的IP范围为202.119.201.65—202.119.201.254。



图5-7 流光v3.3

在全局设置模块，可以设置要扫描的选项：如操作系统信息、弱口令、IIS漏洞等，还可以设置并发扫描的线程的数量，如图5-8所示。

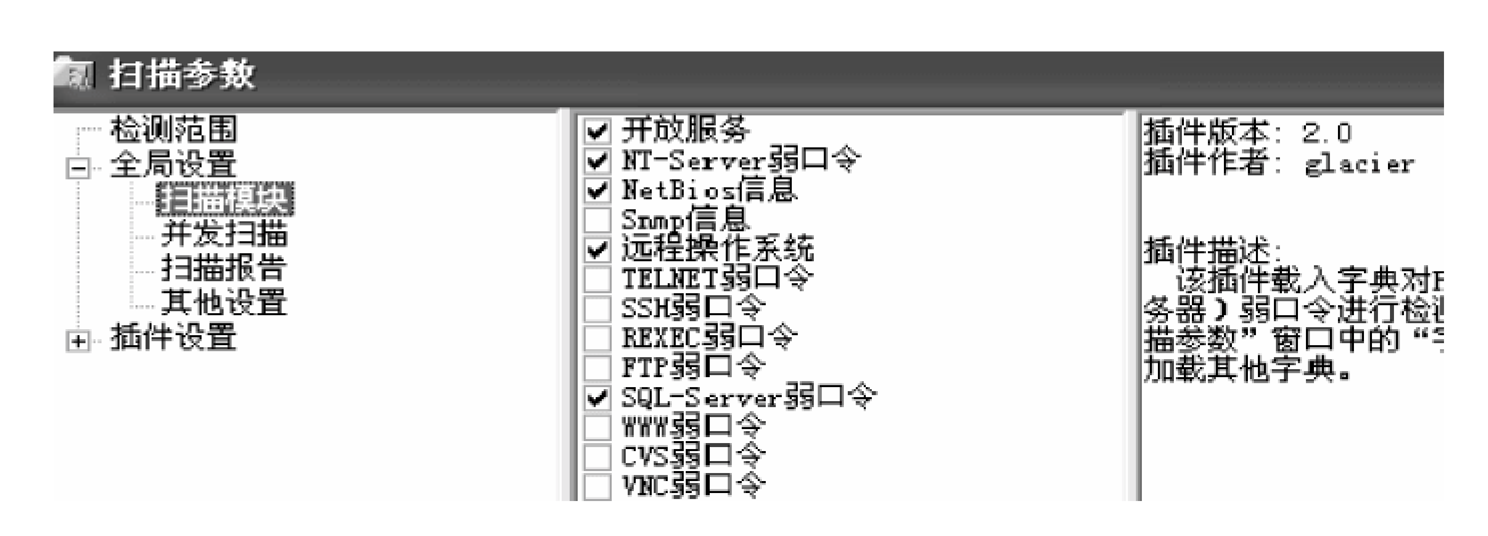


图5-8 扫描参数设置

还可设置扫描插件，如图5-9所示，例如用户可以自己定义字典文件，增加扫描功能。

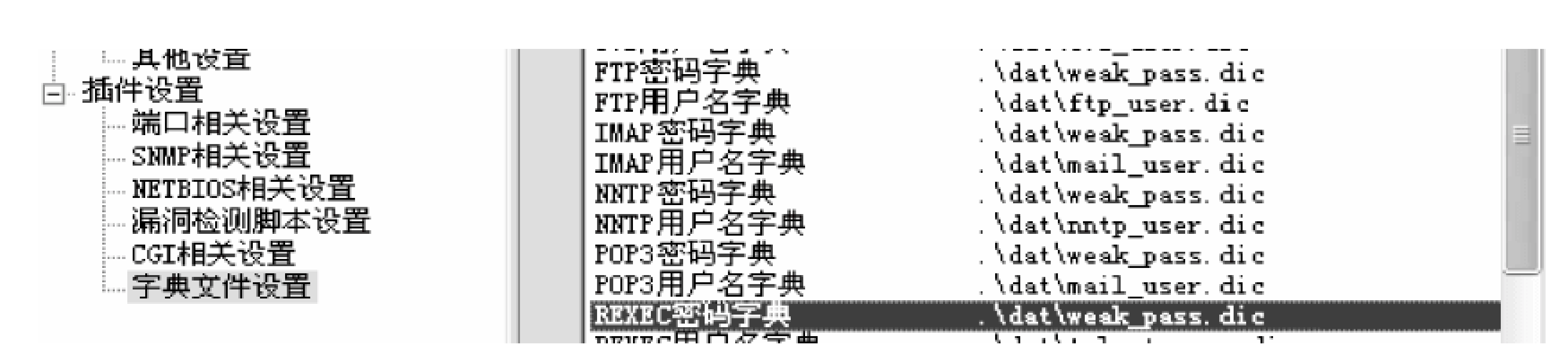


图5-9 设置扫描插件

对刚才设置的网段进行扫描，如图5-10所示。

最终的扫描报告，将以Html文件的形式给出，包含了主机的详细扫描信息和存在的安全问题，如图5-11所示。

5.5.2 Fluxay流光

Fluxay流光不但集成了漏洞扫描、弱口令扫描，而且还附带常用的入侵工具，图5-12为流光V5.0破解版。



图 5-10　扫描设置的网段进行

202.119.201.78	发现安全漏洞
主机摘要 - OS: Unknown OS; PORT/TCP:	

[返回顶部]

主机分析: 202.119.201.78

主机地址	端口/服务	服务漏洞
202.119.201.78	netbios-ns (137/udp)	发现安全提示
202.119.201.78	ms-sql-m (1434/udp)	发现安全漏洞

安全漏洞及解决方案: 202.119.201.78

类型	端口/服务	安全漏洞及解决方案
提示	netbios-ns (137/udp)	**使用NetBIOS探测Windows主机信息** 如果NetBIOS端口(UDP:137)已经打开， 一个远程攻击者可以利用这个漏洞获得主机 的敏感信息，比如机器名，工作组/域名， 当前登陆用户名等。

图 5-11　最终的扫描报告

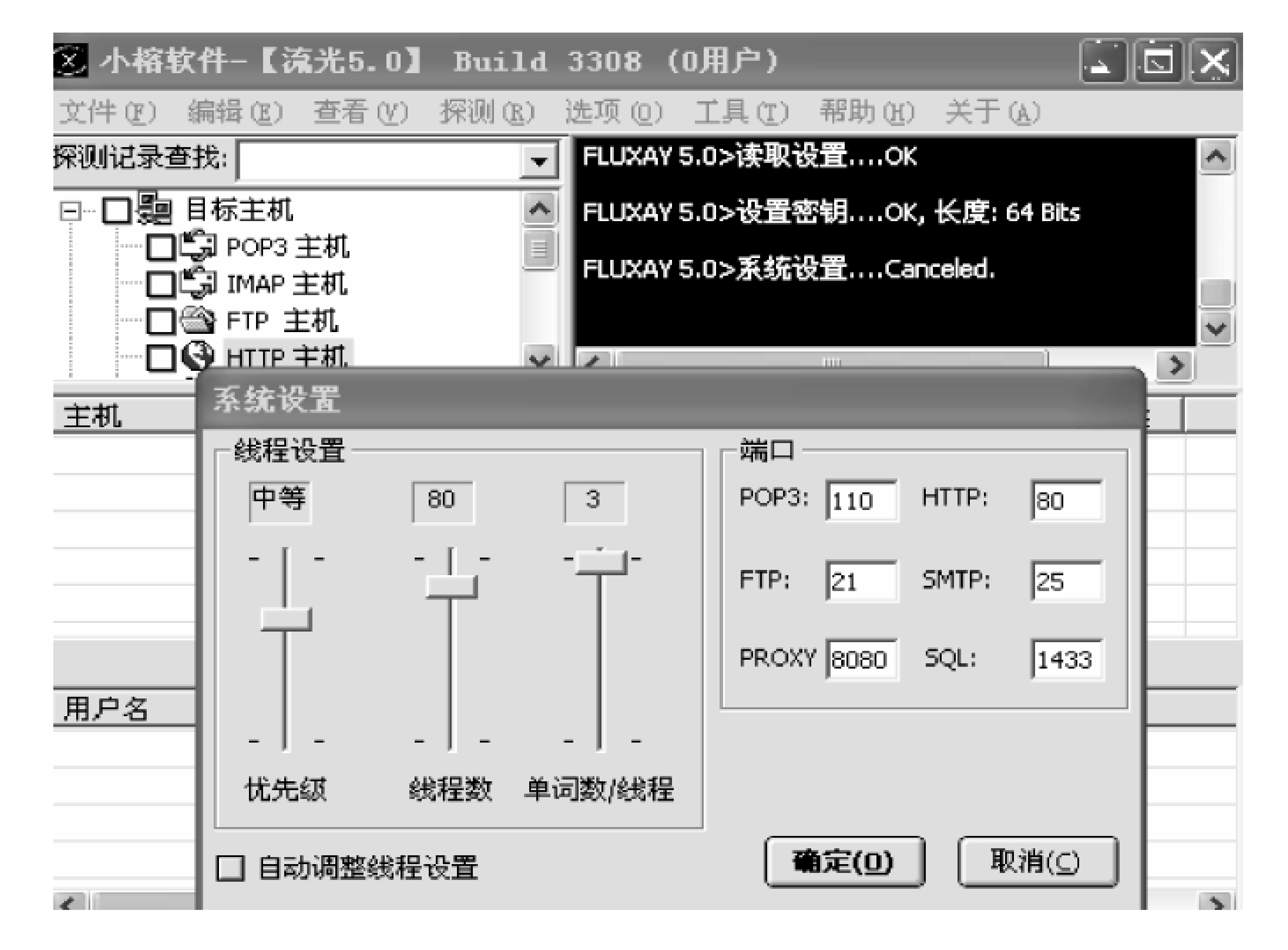


图 5-12　流光 V5.0 破解版

使用流光扫描时，通常要把扫描的线程设置为中等(选择“选项”→“系统设置”命令)，若线程过高，可能会出现错误。流光的功能比 X-Scan 的功能多，而且更复杂，下面介绍两个利用流光扫描的例子。

1. 利用流光扫描 IP 段，检测主机的安全状态

选择“文件”→“高级扫描向导”命令，在弹出的对话框中设置“起始地址”和“结束地址”，选择要检测的项目，如图 5-13 所示。

单击“下一步”按钮，将进入一系列的设置，如“POP”、“Telnet”、“CGI”扫描设置，可根据需要选择或改变默认扫描值，具体设置过程不再叙述。最后一步是设置扫描的主机，在此选择“本地主机”，流光支持“Sensor”扫描，如图 5-14 所示。



图 5-13 流光高级扫描设置



图 5-14 设置扫描的主机

单击“开始”按钮开始扫描，扫描结束后，流光会把扫描的结果以 Html 的形式自动打开，同时在“流光”主界面的下面也给出扫描结果。图 5-15 中扫描到主机 202.119.201.78 的口令为空。

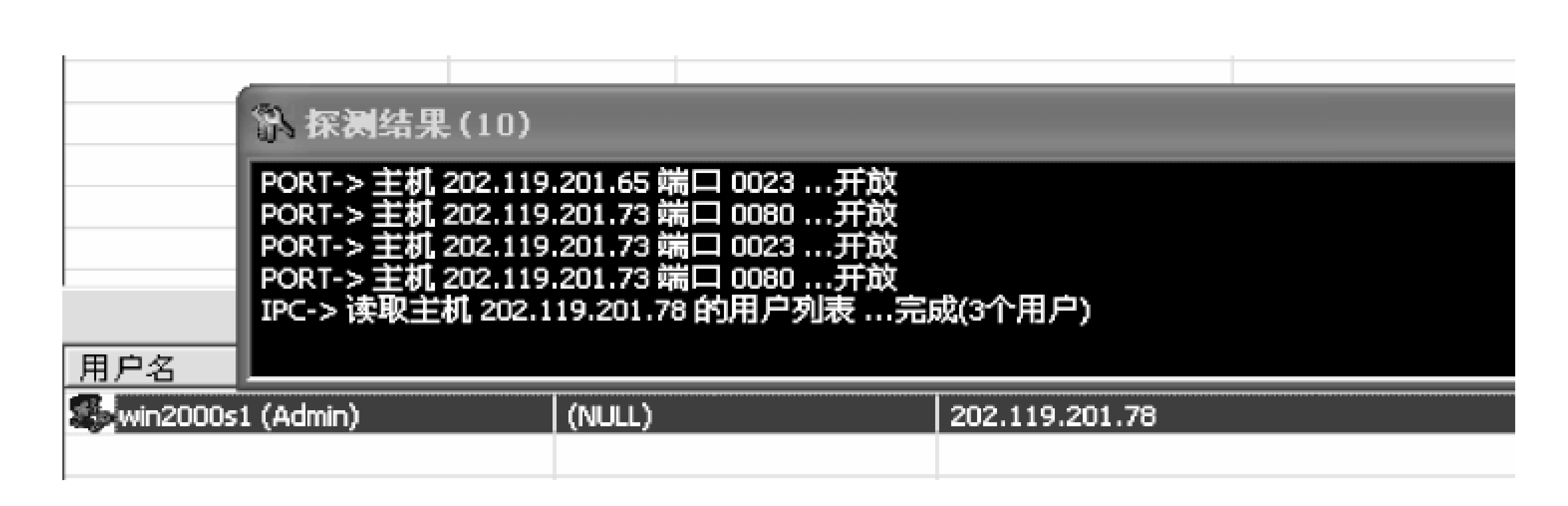


图 5-15 扫描结果

2. 利用流光的“Sensor 扫描”

假设扫描出了主机 202.119.201.78 存在弱口令，单击扫描到的主机 202.119.201.78，在弹出的快捷菜单中选择 “安装 Fluxay Sensor”命令，如图 5-16 所示。

在弹出的安装设置对话框中填写要安装的信息，如图 5-17 所示。系统账号使用的是刚刚扫描到的账号“win2000s1”，该账号必须有管理员权限，安装后的服务名称、显示名称、服务描述等如图 5-17 所示。然后单击“安装”按钮，将会在主机 202.119.201.78 上安装“Fluxay Sensor”。

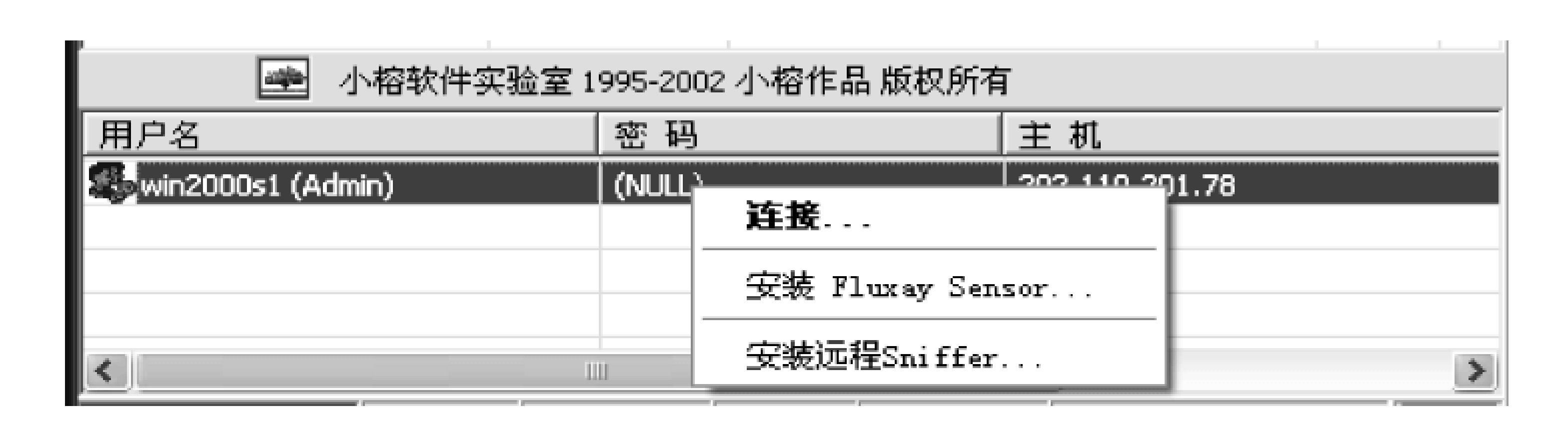


图 5-16 存在弱口令的主机

安装 Fluxay Sensor
主机IP: 202.119.201.78
系统帐号: win2000s1 系统密码:
服务名称: winfluxay
显示名称: winfluxay
服务描述: 流光控制
控制进程: Control 扫描进程: Sensor
注 释:
用 户 名: administrator 控制密码: ***
控制端口: 8080 服务端口: 8080 系统关键性服务 强制覆盖文件
启动服务 winfluxay .....
安装(I) 取消(C)

图 5-17 流光扫描设置

下面利用安装了“Fluxay Sensor”服务的主机 202.119.201.78 进行网段扫描。选择“探测”→“高级扫描设置”命令，弹出的对话框如图 5-18 所示。设置扫描的网段，其他的设置与 4.1.1 设置相同。

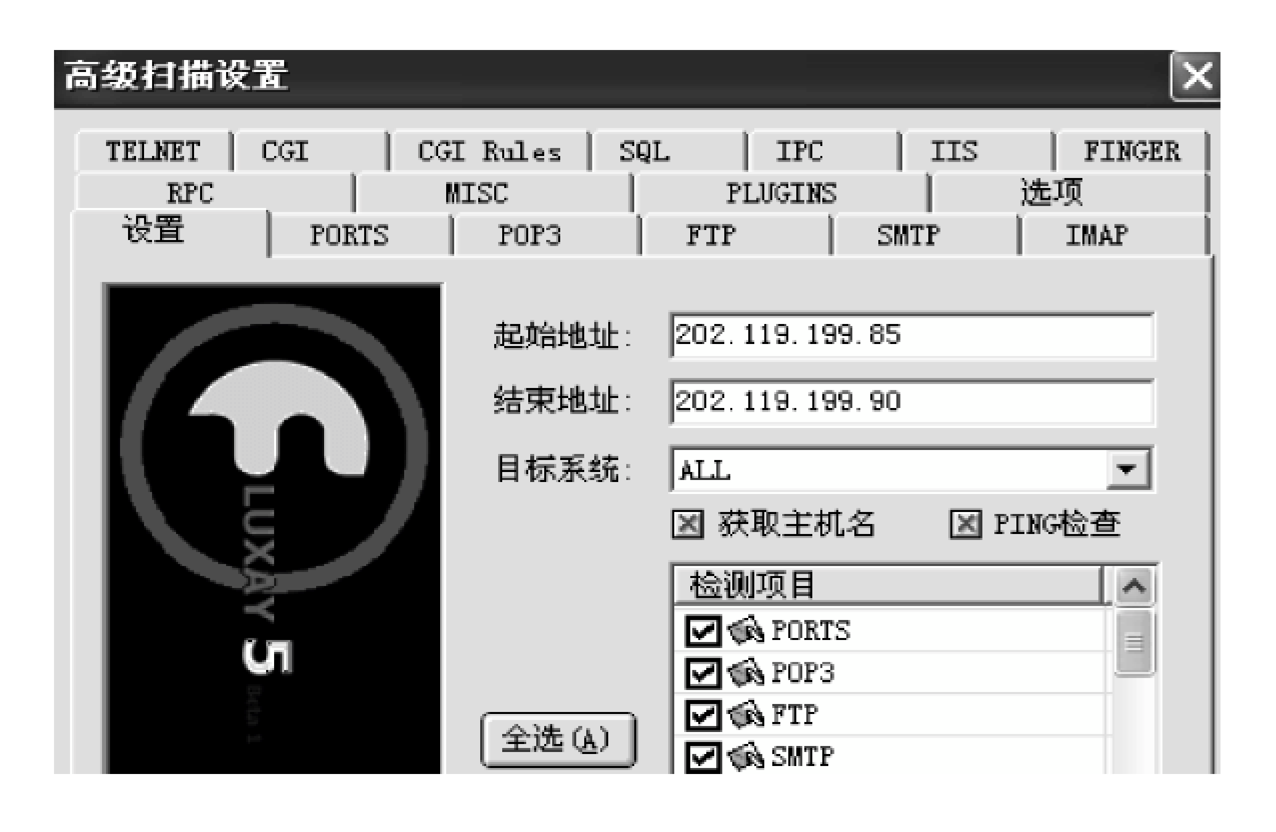


图 5-18 “Fluxay Sensor”服务扫描网段 202.119.201.78

在最后一步“选择流光主机”时，从下拉列表中选择主机 202.119.201.78，然后开始扫描，如图 5-19 所示。

在主机 202.119.201.78 的任务管理器中，可以看到两个进程 Sensor.exe 与 control.exe，如图 5-20 所示，分别是流光安装在 202.119.201.78 上的扫描进程和控制进程。



图 5-19 选择扫描主机

regsvc.exe	1648	00	0:00:00	664 K
mstask.exe	1708	00	0:00:00	2,184 K
Sensor.exe	1752	00	0:00:00	5,880 K
conime.exe	1764	00	0:00:00	1,224 K
VMwareService.e	1772	00	0:00:00	2,420 K
winmgmt.exe	1836	00	0:00:02	264 K
svchost.exe	1936	00	0:00:00	5,824 K
VMwareTray.exe	2068	00	0:00:00	1,952 K
CTFMON.EXE	2076	00	0:00:00	1,908 K
VMwareUser.exe	2092	00	0:00:00	2,092 K
Twister.exe	2104	00	0:00:02	17,572 K
ApacheMonitor.e	2112	00	0:00:00	1,532 K
sqlmangr.exe	2120	00	0:00:00	3,796 K
FilMsg.exe	2132	00	0:00:00	4,388 K
cmd.exe	2136	00	0:00:00	52 K
wuauclt.exe	2408	00	0:00:00	4,576 K
Control.exe	2492	00	0:00:00	1,948 K

图 5-20 扫描进程和控制进程

扫描结果可以看到主机 202.119.199.87 开放了端口 80、23、3306、22、111 以及 CGI 的扫描结果且存在 RPC 远程溢出漏洞。

5.6 扫描器的实现

不管是基于主机的安全扫描还是基于网络的安全扫描,其核心逻辑结构主要由以下部分组成。

1. 策略分析部分

策略分析部分用于控制网络安全扫描器的功能,即它应当检测哪些主机并进行哪些检测。它根据系统预先设定的配置文件决定应当检测哪些域内的主机,并决定对测试目标主机执行的测试级别(简单、中级或高级)。

2. 获取检测工具部分

对于给定的目标系统,获取检测工具部分用于确定对其进行检测的工具。目标系统可以是一个主机,也可以是某个子网上的所有主机(子网扫描)。目标系统可以由用户指定,也可以由分析推断部分根据获取数据部分获得的结果产生。一旦确定了目标系统,获取检测工具部分就可以根据策略分析部分得出的测试级别类,确定需要应用的检测工具,这些检测工具正是获取数据部分的输入。

3. 获取数据部分

对于给定的检测工具,获取数据部分运行对应的检测过程,收集数据信息并产生新的事实记录。安全扫描系统能在检测循环中记录哪些检测是已经执行过的,哪些检测是还未执行的,避免重复工作。最后获得的新的事实记录是事实分析部分的输入。

4. 事实分析部分

对于给定的事实记录,事实分析部分能产生出新的目标系统、新的检测工具和新的事实记录。该部分又分为几个事实分析子过程,每个子过程分别由自己的基本规则集控制,同时该规则集又在子过程的分析中不断更新。新生成的目标系统作为获取检测工具部分的输入,新生成的检测工具又作为获取数据部分的输入,新的事实记录又再一次作为事实分析部分的输入。如此周而复始,直至不再产生新的事实记录为止。

5．报告分析部分

当安全扫描系统执行完网络安全检测之后，会获得关于目标系统的大量信息。报告分析部分则将有用的信息组织起来，用 Html 格式显示，使用户可以通过 Web 浏览器方便查看运行的结果。

安全扫描系统的逻辑结构如图 5-21 所示。

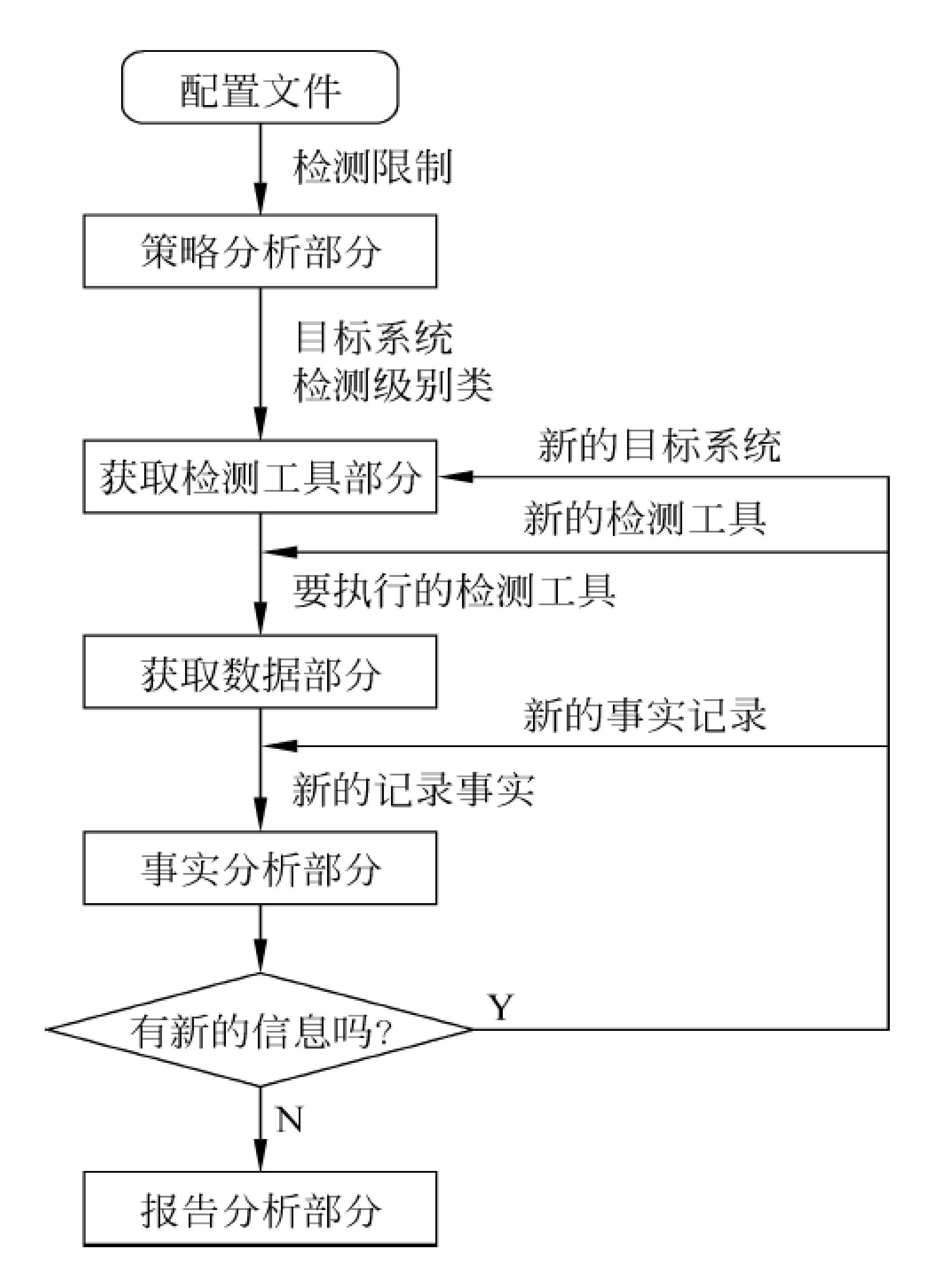


图 5-21　安全扫描器的逻辑结构

习　题　5

1．简单分析系统的安全漏洞。
2．漏洞的分类标准有哪些？
3．漏洞库主要包括哪些信息？
4．分析 CVE 的作用。
5．Windows 的端口扫描方法主要有哪些？
6．说明 TCP Connect()端口扫描服务器端与客户端建立连接的过程。
7．简单列举几个常用的扫描器，并说明它们的工作原理。

第 6 章　攻击与防范

6.1　恶意代码

恶意代码也称为恶意软件，是攻击者植入受害者系统的一段代码，使得攻击者可以在受害者毫不知情的状况下控制对方的系统、网络以及数据。

6.1.1　病毒

计算机病毒是一组能够进行自我传播、需要用户干预来触发执行的破坏性程序或代码。同时，它们还可以把自己的副本分发到其他文件、程序或电脑中去。当被感染文件执行操作的时候(例如：打开一个文件，运行一个程序，点击邮件的附件等)，病毒程序会进行自我复制。

目前，世界范围内已发现的计算机病毒有几万种。尽管这些病毒各自传染的方式不同，发作条件和破坏程度不同，但都具有以下共同特征。

1）非授权可执行性

计算机病毒与其他合法的程序一样，是一段可执行程序，但它又不是一个完整的程序，而是寄生在其他可执行文件上的程序，因此它拥有一切程序所能得到的权利。当病毒运行时，便与合法程序争夺系统的控制权，往往会造成系统崩溃，导致计算机瘫痪。计算机病毒只有在计算机内得以运行时，才具有感染性和破坏性。

也就是说，程序对计算机 CPU 的控制权是关键。若计算机在正常程序控制下运行，而不运行带病毒的程序，则这台计算机是可靠的，在这台计算机上可以查看病毒程序文件的名字、源代码，甚至复制病毒，却不会感染病毒。相反，计算机病毒一旦在计算机上运行，就会与合法程序争夺系统的控制权，表现出病毒症状，造成系统数据丢失或系统崩溃等。

2）感染性

计算机病毒能够将自身的复制品或变种感染到其他程序中。这是计算机病毒的根本特点，它使得计算机病毒得以从被感染的计算机扩散到未被感染的计算机中。计算机一旦感染病毒，如果不及时处理，那么病毒会在这台计算机上迅速扩散，其中大量的文件(主要是可执行文件)会被感染。而被感染的计算机文件又成了新的感染源，可通过各种可能的渠道，如软盘、U 盘和计算机网络等去感染其他的计算机。

3）寄生性

计算机病毒程序是嵌入到宿主程序的，依赖宿主程序的存在而生存，病毒程序侵入到宿主程序以后，一般对宿主程序进行一定的修改，当病毒激活条件一旦满足，病毒程序就被激活，从而可以进行自我复制。

4）隐蔽性

计算机病毒一般都是具有很高编程技巧的程序员编写的短小精悍的程序，通常附着在正常程序中或磁盘较隐蔽的地方，也有个别的以隐含文件形式出现，目的是不让用户发现它的存在。如果不经过代码分析，病毒程序与正常程序是不容易区分开来的。而且受到感染后，计算机系统通常仍能正常运行，使用户不会感到异常。

计算机病毒的隐蔽性表现在两个方面：一是传染过程隐蔽，大部分病毒传染速度快，传染时不具有外表表现，不易被人发觉；二是病毒程序隐蔽，一般病毒程序嵌入在正常程序中，很难被发现。病毒一般只有几百或几千字节，而现在普通 PC 的存取速度都是以 MB/s 为单位，所以病毒在瞬间就可以将其程序附着在正常程序中，使其不容易被察觉。

5）潜伏性

大部分病毒在感染系统后，一般不会马上发作，它可长期隐藏在系统中。除了传染外，不表现出破坏性，只有在满足其特定条件才启动其表现破坏性的模块，显示发作信息或进行系统破坏，只有这样它才可以进行广泛地传播。

6）可触发性

计算机病毒一般都有一个或者几个触发条件。满足其触发条件或者激活病毒的传染机制，使之进行传染；或者激活病毒的表现部分或破坏部分。触发的实质是一种条件的控制，病毒程序可以依据设计者的要求，在一定条件下实施攻击。使计算机病毒发作的触发条件通常有以下几种：利用系统时钟提供的时间作为触发器，这种触发机制被大量的病毒使用；利用病毒体自带的计数器作为触发器；利用计算机内执行的某些特定操作作为触发器。

7）破坏性

无论何种病毒程序一旦侵入系统都会对操作系统的运行造成不同程度的影响。即使不直接产生破坏作用的病毒程序也要占用系统资源（如占用内存空间，占用磁盘存储空间以及系统运行时间等）。根据这些特点，可以将病毒分为良性病毒和恶性病毒：良性病毒程序可能显示一些文字或图像、播放音乐等，没有太大的破坏作用。而恶性病毒将影响系统的正常运行，这些病毒程序删除文件，加密磁盘中的数据，甚至摧毁整个系统和数据，使之无法恢复，造成无可挽回的损失。

8）主动攻击性

病毒对系统攻击是主动的。从一定程度上讲，计算机系统很难彻底地排除病毒对系统的攻击，而保护措施充其量是一种防御的手段。

6.1.2　蠕虫

蠕虫是一种能在不需要用户干预的情况下即可触发，并自动传染计算机的破坏性程序或代码。蠕虫不修改文件，而是常驻在内存里并复制自己。蠕虫使用操作系统的一部分，这部分对于用户来说是自动且无形的。通常只有在它的无法控制的疯狂复制占用了系统资源，使得其他的任务缓慢甚至停滞的情况下才会发现它们。蠕虫的自我复制不像其他的病毒，它可以自动创建与它的功能完全相同的副本，并在没人干涉的情况下自动运行。蠕虫是通过系统存在的漏洞和系统设置的不安全性（例如：设置共享）来进行入侵的。它的自身特性可以使它以很快的速度传输（在几秒钟内从地球的一端传送到另一端）。

计算机病毒和蠕虫有很多的相似之处，例如都可进行自我复制，感染破坏系统。但它们

之间有很大的区别。

(1) 蠕虫是一个可以独立运行的程序。但病毒不能独立存在,必须将自身的代码附着在其他程序中,其程序的激活依赖于宿主程序的执行。

(2) 蠕虫可以自动通过传播,不需要利用文件寄生技术;而病毒要依赖于宿主文件进行传播。因为蠕虫程序自身包含了传播代码,这些代码会自动将蠕虫程序从一个系统发送到另一个系统。例如,蠕虫可以自动将自己包含在邮件附件中发送给其他用户;而病毒必须是用户将感染病毒的文件作为附件发送时,才能感染新的系统。

(3) 蠕虫通常感染的对象是有相应漏洞或者其他脆弱性的计算机系统,而病毒的感染对象则是计算机中的文件系统。

表 6-1 是计算机病毒与蠕虫之间的区别。

表 6-1 蠕虫与病毒的区别

名　　称	计算机病毒	蠕　　虫
存在形式	寄生于其他程序	独立程序
感染目标	本地文件	网络中的计算机
传播途径	感染文件或可移动磁盘	网络传播
传播形式	需要人为干预,例如,运行感染病毒的程序,使用感染病毒的磁盘	不需要人为干预,蠕虫会利用系统漏洞或系统脆弱性、电子邮件等将自身副本发送到其他系统

6.1.3 木马

特洛伊木马从表面上看是正常的程序、具有正常的功能,但是实际上却隐藏着带有恶意目的的程序。一些木马程序会通过覆盖系统中已经存在的文件的方式存在于系统之中,同时它可以携带恶意代码,还有一些木马会以一个软件的身份出现(例如:一个可供下载的游戏),但它实际上是一个窃取密码的工具。这种木马程序通常不容易被发现,因为它一般是以一个正常的应用的身份在系统中运行的。

特洛伊木马通常由一个客户端和一个服务器端构成,客户端放在木马控制者的电脑中,服务器端放置在被入侵电脑中,木马控制者通过客户端与被入侵电脑的服务器端建立远程连接。一旦连接建立,木马控制者就可以通过对被入侵电脑发送指令来传输和修改文件。还有一些木马不具备远程登录的功能。它们的存在只是为了隐藏恶意进程的痕迹,例如使恶意进程不在计算机的进程列表中显示出来。还有一些木马用于收集信息,例如被感染电脑的密码,并把收集到的密码列表发送互联网中一个指定的邮件账户中。常见的木马有冰河、网络神偷、灰鸽子。

常见的木马可以分为以下几类。

1) 远程访问型

这是目前使用最广的特洛伊木马。这类木马可以远程访问被攻击者的硬盘。

远程访问型特洛伊木马会在目标计算机上打开一个端口。一些特洛伊木马还可以改变端口的选型并且可以设置连接密码,为的是只能让攻击者来控制特洛伊木马。改变端口的选型是非常必要的,因为一些常见木马的监听端口已经为广大用户熟知了。新的远程访问

型特洛伊木马每天都在出现。

2）密码发送型

这种特洛伊木马的目的是找到所有的隐藏密码，并且在被攻击者不知道的情况下把它们发送到指定的信箱。这类特洛伊木马大多数会在每次 Windows 重新启动的时候运行，而且它们大多使用 25 号端口发送 E-mail。

3）键盘记录型

这种特洛伊木马非常简单：它们只做一种事情，就是记录受害者的键盘输入并且在 log 日志文件中查找密码。这种特洛伊木马随着 Windows 系统的启动而运行。它们有记录在线和离线的功能。在线选项中，它们知道被攻击者在线并且记录每一件事情。但是，在离线记录时，每一件事情在 Windows 启动后才被记录，并且保存在受害者磁盘上等待被传输。

4）毁坏型

这种特洛伊木马的唯一功能是毁坏并且删除文件。它们非常简单，也很容易被发现。它能自动删除目标计算机的所有后缀名如 dll、ini、exe 等文件，所以非常危险，一旦被感染了，就会严重威胁到计算机的安全。

5）FTP 型

这类特洛伊木马程序打开目标计算机的 21 号端口，使黑客可以用一个 FTP 客户端并且不用密码就可以连接到目标计算机上，并且拥有完全的上传和下载的权限。

6.1.4　后门

后门通常指房间后面的可以自由出入的门，是相对于明显的前门而言的。对计算机操作系统或软件来说，是指可以绕过软件的安全性控制而从比较隐秘的通道获取对程序或系统访问权的黑客方法。通常，在软件开发时，设置后门可以方便修改和测试程序中的缺陷。但如果后门被其他人知道（可以是泄密或者被探测到后门），或是在发布软件之前没有去除后门，那么它就对计算机系统安全造成了威胁。

后门具有隐蔽性，能绕开系统日志，不易被系统管理员发现等特点。

在计算机中，入侵者可以通过端口、串/并口、无线设备连接等后门方式进行入侵。如果一个程序仅仅提供远程访问，那么它只是一个后门。如果攻击者将这些后门伪装成某些其他良性程序，那么就变成了特洛伊木马。

后门产生的必要条件有以下三点：

（1）必须以某种方式与其他终端节点相连。由于后门的利用都是从其他节点进行访问，因此必须与目标机使用双绞线、光纤维、串/并口、蓝牙、红外等设备在物理信号上有所连接才可以对端口进行访问。只有访问成功，双方才可以进行信号交流，攻击方才有机会进行入侵。

（2）目标机默认开放的可供外界访问的端口必须在一个以上。因为一台默认无任何端口开放的机器是无法连接通信的，而如果开放着的端口外界无法访问，则同样没有办法进行入侵。

（3）目标机存在程序设计或人为疏忽，导致攻击者能以权限较高的身份执行程序。并不是任何一个权限的账号都能够被利用，只有权限达到操作系统一定要求的才允许执行修改注册表，修改 log 记录等相关修改。

通常后门产生的原因有以下几种。

(1) 操作系统自带的服务，大多数的操作系统，如 UNIX、Linux、Windows 2000/XP 在默认安装完成后，为了方便用户的使用，会默认开启一定数量的服务(server)，常见的有 telnet server、ssh server、sendmail 等，如果有的服务程序版本存在漏洞或者默认配置的安全性不够，则很容易被攻击，从而成为攻击者进入的通道。

(2) 早期网络协议安全性问题。由于早期互联网络是用于军方、高校及科研结构，网络协议的设计对安全性方面考虑较少。例如，目前所使用的 IPv4 协议，IP 数据包的源地址可以被修改、MAC 地址可以被伪造。

以 NetBIOS 为例：NetBIOS 主要是用来在网络上鉴别资源的，程序使用这些名称开始和结束会话。这个协议的特点是简单高效，缺点则是不验证对方身份就可以获得信息。因此对于安全验证方面相对薄弱，许多病毒都是通过共享途径传播的，例如爱情后门、欢乐时光等。一旦机器感染病毒，就可能为攻击者留下后门。

(3) 软件编写不规范。现在很多大型软件都是模块化编程，可能某些功能模块之间出现漏洞，被攻击者利用作为漏洞；也有可能是程序员作为程序测试用的通道没有去除，被攻击者发现。

6.2 网络嗅探

网络监听也称为网络嗅探，是一种发展比较成熟的技术，通常网络管理员可以利用网络监听工具监测网络状态、排除网络故障。然而，网络监听也给以太网带来了巨大的安全隐患，网络监听工具成了黑客的常用工具，黑客通过将网络接口设置成监听模式，便可截取以太网中的信息，造成用户口令失窃、敏感数据泄漏等。

网络监听可以在网上的任何一个位置，如局域网中的一台主机、网关或远程网中的调制解调器等。监听效果好的是在网关、路由器、防火墙一类的设备上，通常由网络管理员来操作。黑客用的最多的是用网络监听来截取用户口令。

6.2.1 共享式以太网监听

当局域网内的主机是通过集线器(HUB)等方式连接时，一般采取的是共享的连接。这种共享方式连接的特点是，源主机的网络接口卡将要发送的数据报添加目的 MAC 地址、源 MAC 地址等信息封装成帧，然后将帧以广播的方式发往局域网。当一台主机向另一台主机发送数据时，共享式的 HUB 会将接收到的数据向 HUB 上的每个端口转发。所以，从理论上说，当采用共享式连接时，位于同一网段的每台主机都可以截获在网络中传输的所有数据。

通常情况下只有与数据帧中目的 MAC 地址一致的主机才会接收数据，并对其进行处理。网卡对于不属于自己的报文不予以响应，只是简单的忽略掉这些数据。但是，如果某台主机将其网卡设置为混杂模式，使其工作在监听模式下，则不管数据帧中的目标地址是什么，主机都可以将其捕获。

那么，当主机工作在监听模式下，将接收到的所有数据帧交给上层协议软件处理。而

且,当连接在同一条电缆或集线器上的主机被逻辑地分为几个子网时，如果一台主机处于监听模式下,它还能接收到发向与自己不在同一子网(使用了不同的掩码、IP 地址和网关)的主机的数据包。也就是说,在同一条物理信道上传输的所有信息都可以被接收到。另外,现在网络中使用的大部分协议都是很早设计的,许多协议的实现都是基于一种非常友好的、通信的双方充分信任的基础之上，许多信息以明文发送。因此,如果用户的账户名和口令等信息也以明文的方式在网上传输,而此时一个黑客或网络攻击者正在进行网络监听,只要具有初步的网络和 TCP/IP 协议知识,便能轻易地从监听到的信息中提取出感兴趣的部分。同理,正确的使用网络监听技术也可以发现入侵并对入侵者进行追踪定位,在对网络犯罪进行侦查取证时获取有关犯罪行为的重要信息,成为打击网络犯罪的有力手段。

6.2.2　交换式以太网监听

随着以太网交换机的普及,现在大部分的局域网是交换式以太网。以太网交换机的原理很简单,它检测从以太网主机传送到端口来的数据包的源主机和目的主机的 MAC(介质访问层)地址,然后再与系统内部的动态查找表进行比较,发往目的主机对应的交换机端口;若数据包的 MAC 地址不在查找表中,则将该地址加入查找表中,并将数据包发送给相应的目的端口。

由于交换式以太网的交换机能够将数据报文准确地发送给数据报文的目的主机,而不会同时发给其他的计算机。因此,在交换网络的环境下,实现数据包的监听要复杂一些。

目前,实现交换式以太网的数据包监听的方法主要有以下几种。

(1) 对交换机实行端口映射,将该端口的数据包全部映射到某个监控机器上。

(2) 将数据包捕获程序放在网关或代理服务器上,这样就能抓取到整个局域网的数据包。

(3) 在交换机和路由器之间连接一个 HUB,这样数据将以广播的方式发送。

(4) 实行 ARP(地址解析协议)欺骗,即在用户机器上实现整个包的转发,但这样会降低整个局域网的效率。

下面以 ARP 欺骗为例子,分析交换环境下的网络监听的原理。

ARP 协议的全称是地址解析协议,ARP 协议的功能是把逻辑地址转换为物理地址,即将 32 位的 IP 地址转换成 48 位的物理地址(网卡的 MAC 地址)。例如,一台 IP 地址为 192.168.0.1 的计算机,其网卡的 MAC 地址为 00-03-0F-FD-1D-2B。转换过程是一台主机向局域网广播包含目标 IP 地址的数据包,即 ARP 请求,然后目标主机向该主机发送一个含有其 IP 地址和 MAC 地址的数据包,即 ARP 应答。

当一个网络设备需要和另一个网络设备通信时,它首先把目标设备的 IP 地址与自己的子网掩码进行“与”操作,以判断目标设备与自己是否位于同一网段。如果在同一个网段,而且源设备没有在高速缓存中获得与目标 IP 地址相对应的 MAC 地址信息,则源设备以广播的形式发送 ARP 请求报文,在 ARP 请求报文中包含了源设备与目标设备的 IP 地址。ARP 报文以广播的形式在网络中发送,因此同一网段中的所有其他设备都可以收到并分析这个 ARP 请求报文。若某设备接收到 ARP 报文并发现报文中的目标 IP 地址与自己的 IP 地址相同,则向源设备发回 ARP 响应报文,通过该报文使源设备获得目标设备的 MAC 地址信息。如果目标设备与源设备不在同一个网段,则源设备首先将 IP 分组发向自己的默认

网关,由默认网关对该分组进行转发。若源设备的缓存中没有默认网关的 MAC 地址,则它将同样以发 ARP 广播包的方式来获取网关的 MAC 地址。

也就是说无论与源设备通信的目标设备是否在同一个网段,数据包在局域网中都要和另一台设备直接通信。而在以太网中,一个网络设备要和另一个网络设备进行直接通信,需要知道目标设备的 MAC 地址。

在以太网里,为了减少广播量,且避免频繁发送 ARP 请求包进行地址转换,每台主机都有一个 ARP 高速缓存,其中记录了最近一段时间内与其通信的设备的 IP 地址和 MAC 地址的映射信息。如果本机想与某台主机通信,则首先在 ARP 缓存中查找这台主机的 IP 和 MAC 信息,若存在,则直接利用此 MAC 地址构造以太网数据帧;若不存在,则向网络上广播一个 ARP 请求包。目的主机收到此请求包后,发送一个 ARP 应答包。本机收到此应答包后,把相关信息记录在 ARP 高速缓存中,然后再进行发送。

然而,对大多数操作系统而言,如果收到一个 ARP 应答,它们不管自己是否在此之前曾发出 ARP 请求报文,都会自动更新自己的 ARP 缓存,而不对 ARP 应答的真实性进行验证,因此 ARP 是一个无状态的协议,ARP 欺骗就是利用这一点进行欺骗。

ARP 欺骗的核心思想就是向目标主机发送伪造的 IP-MAC 映射的 ARP 响应,使目标主机收到伪造的 ARP 应答后更新其 ARP 缓存,从而使目标主机将数据包发送给指定的主机。

假如同一交换网络中的三台主机 A、B、C,主机 C 试图通过 ARP 欺骗截取主机 A 发送给主机 B 的数据报文,达到监听 A、B 通信的目的。

欺骗过程步骤为:

(1) 主机 C 发送 ARP 请求报文,分别获取主机 A 和 B 的 MAC 地址。

(2) 主机 C 向主机 A 发送伪造的 ARP 应答报文(报文的源 IP 地址为 B 的 IP 地址,源 MAC 地址为 C 的 MAC 地址)。

(3) 主机 A 收到 ARP 应答报文,根据 C 发送的伪造的 ARP 应答报文,更新 ARP 缓冲区,此时有一条 IP-MAC 对应关系:主机 B 的 IP 地址—主机 C 的 MAC 地址。

那么当主机 A 的应用程序向主机 B 发送数据时,在网络层形成包含主机 B 的 IP 信息的 IP 数据包;而在数据链路层将主机 C 的 MAC 地址信息加在 IP 数据包前,构成以太网数据帧,发往交换机的端口。交换机查找其主机 MAC 地址与端口的对应关系,将数据帧发往主机 C 对应的端口。

(4) 主机 C 将接收的主机 A 发往主机 B 的数据帧去掉头部,加上主机 B 的 MAC 地址信息,发往交换机端口。交换机根据端口与 MAC 对应关系,将数据发往主机 B 对应的端口。

这样,主机 C 成功的截获主机 A 发往 B 的信息,而 A 与 B 并未发觉通信有任何异常。同样主机 C 也可以欺骗主机 B,截获 B 发往 A 的数据。这样主机 C 就可以监听 A 与 B 之间的所有通信。这种攻击通常也称为中间人攻击。

为了保证伪造的 IP-MAC 映射关系在主机 A 缓冲区中的有效性,主机 C 每隔一个时间片发送一次 ARP 应答报文,保证主机 A 处于被欺骗状态。

同样,主机 C 也可以监听主机 A 或 B 与外网的通信,则只需要将上述步骤中的 A 或 B 中的一个换为网关即可。

可知,通过 ARP 欺骗,可监听交换网络中的通信。

6.3　缓冲区溢出

1988 年 11 月,许多组织与机构不得不因为“Morris 蠕虫”而切断 Internet 连接,“Morris 蠕虫”是 23 岁的程序员 Robert Tappan Morris 编写的用于攻击 VAX 和 Sun 机器的程序。据有关方面估计,这个程序大约使得整个 Internet 的 10%崩溃。Morris 蠕虫利用了一个被称为缓冲区溢出的程序缺陷。

6.3.1　缓冲区溢出原理

缓冲区溢出的原理如图 6-1 所示,由于内存的生长方向与堆栈的生长方向是相反的,堆栈是从内存的低端向内存的高端生长,那么向堆栈中压入的数据超过了堆栈预先分配的容量时,就会覆盖内存高端的有用数据,此时就会出现堆栈溢出,从而使得程序失败,如果发生堆栈溢出的是大型程序,则有可能导致系统崩溃。

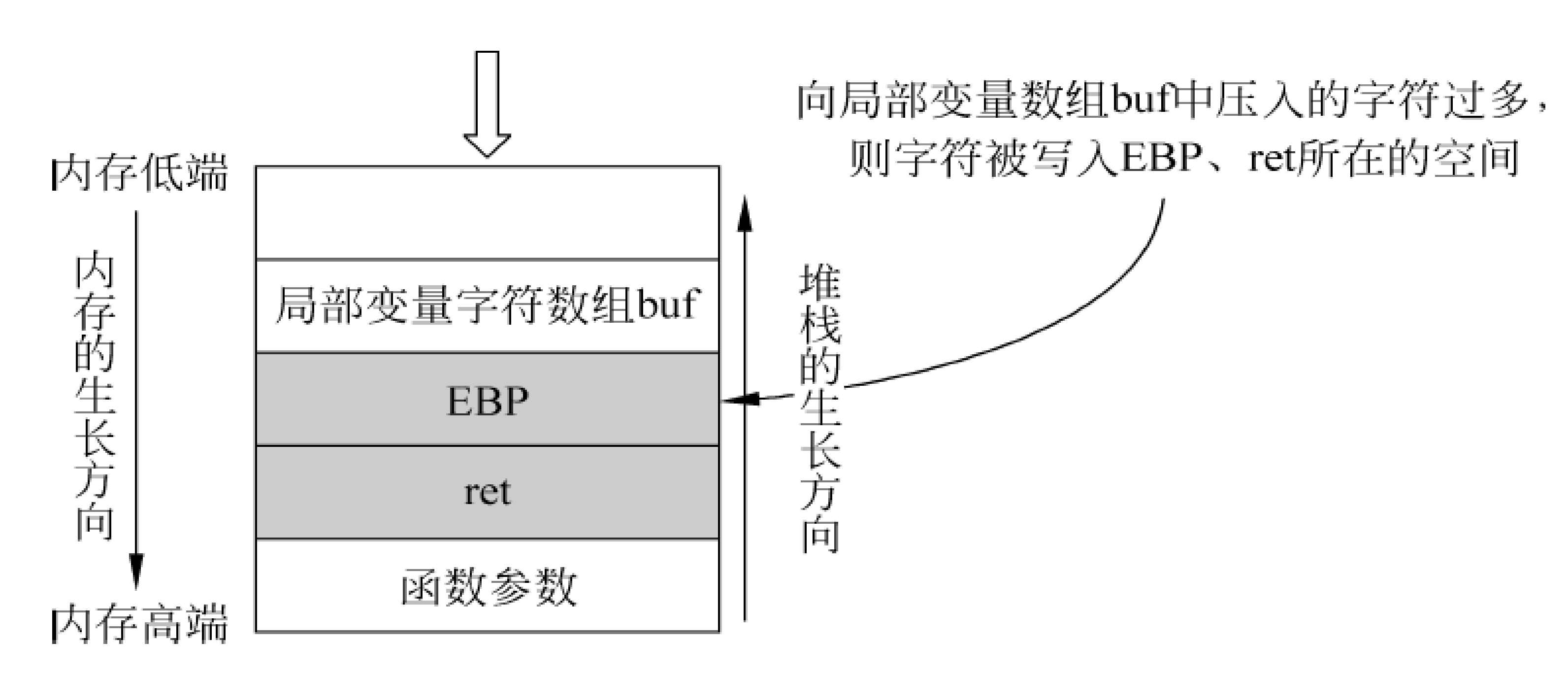


图 6-1　缓冲区溢出的原理

如果仅仅是因为如例子中的,输入字符串的长度超过了预先分配的容量,覆盖了 EBP、ret 的空间,那么可能造成的损害是比较小的。如果攻击者编写黑客程序,通过覆盖有缓冲区漏洞的程序的返回地址,让函数返回后直接执行他们编写的代码,将造成巨大的损失。

6.3.2　缓冲区溢出实例

这里主要介绍一下 Microsoft Windows Messenger 服务远程堆溢出漏洞的情形。

Windows Messenger 服务用于 Microsoft Windows 操作系统服务器与客户端之间互相发送一些短消息。Microsoft Windows Messenger 服务存在堆溢出问题,远程攻击者可以利用这个漏洞以系统权限在目标机器上执行任意指令。问题存在于 Messenger 服务程序的 search-by-name 函数中,攻击者提交特定序列的字符串给这个函数可造成堆溢出,精心构建提交数据可能以系统权限在目标机器上执行任意指令。消息通过 NetBIOS 或者 RPC 提交给消息服务,因此可以通过封闭 NETBIOS 端口(137-139)和使用防火墙过滤 UDP 广播包来阻挡此类消息。

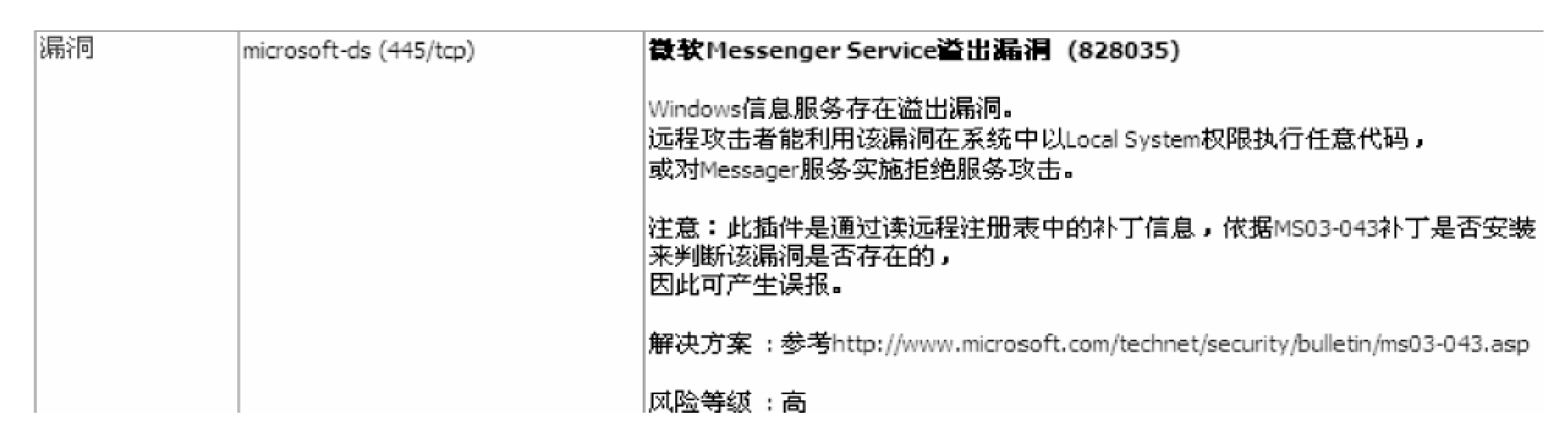
用到的工具有 X-Scan 3.3、专用于 Microsoft Windows Messenger 服务远程堆溢出漏洞的扫描工具 RetinaMSGSVC.exe、漏洞利用工具 msgdos.exe。

首先,对目标主机进行漏洞扫描,利用 X-Scan3.3 对 202.119.201.1～202.119.201.254 网段进行漏洞扫描,发现主机 202.119.201.71 存在安全漏洞,如图 6-2 所示。

202.119.201.71	smtp (25/tcp)	发现安全漏洞
202.119.201.71	tcp	发现安全漏洞

图 6-2 发现主机漏洞

其中的一个漏洞为 Microsoft Windows Messenger 服务远程堆溢出漏洞,图 6-3 为漏洞信息。



漏洞	microsoft-ds (445/tcp)	**微软Messenger Service溢出漏洞 (828035)** Windows信息服务存在溢出漏洞。 远程攻击者能利用该漏洞在系统中以Local System权限执行任意代码， 或对Messager服务实施拒绝服务攻击。 注意：此插件是通过读远程注册表中的补丁信息，依据MS03-043补丁是否安装 来判断该漏洞是否存在的， 因此可产生误报。 解决方案 ：参考http://www.microsoft.com/technet/security/bulletin/ms03-043.asp 风险等级 ：高

图 6-3 漏洞描述

为确定目标主机 202.119.201.71 确实存在该漏洞,利用 Microsoft Windows Messenger 服务远程堆溢出漏洞的扫描工具 RetinaMSGSVC.exe 对其进行扫描(如图 6-4 所示),扫描结果出现 VULENRABLE to MS03-43,说明该主机确实可能存在此漏洞。

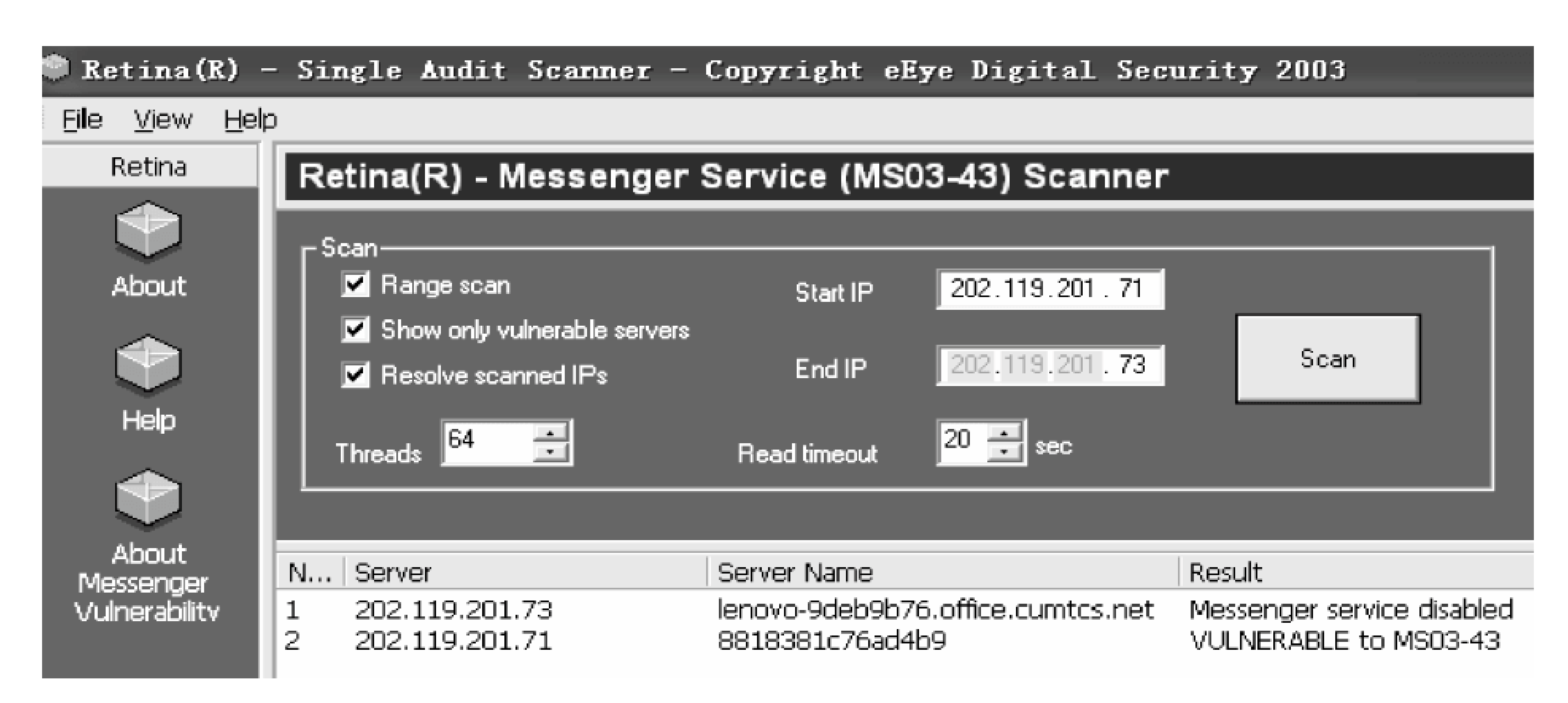


图 6-4 RetinaMSGSVC.exe 扫描结果

然后,利用 msgdos.exe 对主机 202.119.201.71 进行攻击,msgdos.exe 是一个命令行工具,如图 6-5 所示。

```
F:\>msgdos.exe 202.119.201.71
Max 'body' size (incl. terminal NULL char) = 3940
Total length of strings = 3948
Packet size = 4072
Fields size = 3992
```

图 6-5 利用 msgdos.exe 对主机进行攻击

如果溢出成功,则目标主机将会在一分钟后重启,如图 6-6 所示。



图 6-6　目标主机被溢出

6.3.3　缓冲区溢出的防范

针对缓冲区溢出，程序员和系统管理员应采取以下一些措施防御缓冲区溢出。

1. 编写正确的代码

缓冲区溢出漏洞存在的原因是在程序开发时使用了与字符串操作相关的函数，没有对函数参数进行严格控制，导致数据越界造成的。如果能确保缓冲区中数据不越界并有效，就可以避免缓冲区溢出。由于 C/C++语言注重效率而忽视正确性，是不进行强类型和长度检查的一种程序设计语言，程序员在编写代码时由于开发速度和代码的简洁性，往往忽视了程序的健壮性，从而导致缓冲区溢出。

很多存在漏洞的程序是由于调用了一些不安全的库函数，这些库函数往往没有对数组边界进行检查。现在已经开发了一些工具和技术来帮助程序员编写安全正确的代码，例如利用 grep 搜索源代码中容易产生漏洞的库的调用。

2. 非执行的缓冲区技术

所谓的非执行的缓冲区技术，也称为堆栈不可执行，就是指通过使被攻击程序的数据段地址空间不可执行，从而使得攻击者不可能执行被植入被攻击程序输入缓冲区的代码。但是，不可能使得所有程序的数据段不可执行，UNIX 和 MS Windows 系统为实现更好的性能和功能，需要在数据段中动态地放入可执行的代码。所以为了保持程序的兼容性不可能使得所有程序的数据段不可执行，但是可以设定堆栈数据段不可执行，这样就可以最大限度地保证了程序的兼容性。

非执行的缓冲区技术可以有效地对付把代码植入自动变量的缓冲区溢出攻击，而对于其他形式的攻击则没有效果。通过引用一个驻留的程序的指针，就可以跳过这种保护措施。攻击者也可以采用把代码植入堆或者静态数据段中来跳过保护。

3. 数组边界检查

缓冲区溢出的一个重要原因就是没有对数组进行边界检查，如果数组不能被溢出，也就不可能发生数组溢出攻击。当数组被溢出的时候，函数返回地址、过程帧指针、函数指针等重要数据就有可能被修改，被植入的攻击代码就可能执行。因此，对数组进行边界检查，使

超长代码不可能植入，这样就完全没有了缓冲区溢出攻击产生的条件。对数组边界检查，直接的办法就是对所有的数组读写操作都进行检查，但这种完全检查要付出很大的性能代价。

4. 程序指针完整性检查

程序指针完整性检查是指在程序指针被应用之前检测到指针的改变，因此即使一个攻击者成功地改变程序指针，由于系统提前检测到了指针的改变，因此这个指针将不会被使用。程序指针完整性检查通过阻止缓冲区溢出的一个条件——由于函数返回地址或函数指针的改变而导致的程序执行流程的改变，来阻止缓冲区溢出的发生。

它的原理是每次在程序指针被引用之前先检测该指针是否已被恶意改动过，如果发现被改动，程序就拒绝执行。因此，即使一个攻击者成功地改变程序的指针，由于系统事先检测到了指针的改变，因此这个指针不会被使用。常用的程序指针完整性检查有堆栈保护和指针保护。

堆栈保护是一种提供程序指针完整性检查的编译器技术，即在堆栈中函数返回地址后加了一些附加的字节，函数返回时，如果检测到这个附加的字节被改动过，则说明发生过缓冲区溢出攻击。但是如果攻击者预见到这些附加字节的存在，便可以在溢出过程中同样地附加字节，此时可以利用终止符号和随机符号对付这种攻击，由于终止符号不能在常用的字符串函数中使用，这些函数一旦遇到这些终止符号，函数过程就结束了，随机符号方式使得攻击者不可能猜测到附加字节的内容，从而加大攻击难度。

指针保护是堆栈保护的一个推广，通过在所有的代码指针之后放置附加字节来检验指针在被调用之前的合法性，如果校验失败，会发出警报信号和退出程序的执行。当然，正如在堆栈保护中的行为一样，程序指针完整性检查也不能解决所有的缓冲区溢出问题。

5. 及时安装安全补丁

管理员必须不断关注最新的技术和补丁，不断地对系统进行修补，这是漏洞出现后最迅速有效的补救措施。大部分的入侵是利用一些已被公布的漏洞达成的，如能及时补上这些漏洞，将增强系统抵抗攻击的能力，有效地防止大部分的攻击。

6. 关闭不需要的特权程序

由于缓冲区溢出只有在获得更高的特权时才有意义，所以带有特权的 UNIX 下的 suid 程序和 Windows 下由系统管理员启动的服务进程都经常是缓冲区溢出攻击的目标。这时候，关闭一些不必要的特权程序就可以降低被攻击的风险。如 Solaris 下的 fdformat 是个有缓冲区溢出漏洞的 suid 程序，因为这个格式化软盘的命令用得较少，最直接的措施是去掉这个程序或者去掉 suid 位。当有缓冲区溢出漏洞的程序还没有补丁时，就可以用这种方法。

6.4 SQL 注入

SQL 注入是漏洞很常见的一种类型，一般来说，当非信任数据被插入一条查询或指令时，指令注入漏洞就会出现。恶意输入会导致指令解释器或查询处理器对提供的指令数据产生误解，反之改变指令或查询原来的语义。另外，对于 SQL 查询而言，如果应用程序执行 shell 指令、查询 LDPA 服务器、使用 XPath 语句从 XML 文件中抽取数据、把非信任数据翻

译为 XSLT 样式单中的一部分等，那么这个问题将会出现。

对通过 Google 搜寻找到的 1000 个网站的随机取样测试，检测到其中的 11.3%有易受 SQL 注入攻击的可能。这是很可怕的，这意味着黑客可以远程利用那些应用里的数据，获取任何没有 hashed 或加密的密码或信用卡数据，甚至有以管理员身份登录进这些应用的可能。这不仅对开发网站的开发人员来说很糟糕，而且对使用网站的消费者或用户来说更糟糕，因为他们给网站提供了数据，认为网站是安全的。

6.4.1 SQL 注入攻击的原理

SQL 注入攻击源于英文"SQL Injection Attack"，目前还没有看到一种标准的定义，常见的是对这种攻击形式、特点的描述。微软技术中心从两个方面进行了描述：脚本注入式的攻击；恶意用户输入导致被执行的 SQL 脚本受到影响。Stephen Kost 给出了这种攻击形式的另一个特征："从一个数据库获得未经授权的访问和直接检索，当攻击者能够操作数据，往应用程序中插入一些 SQL 语句时，SQL 注入攻击就发生了。"SQL 注入攻击就其本质而言，利用的是 SQL 的语法，针对的是应用程序开发者编程过程中的漏洞。

由于 SQL 注入攻击使用的是 SQL 语法，使得这种攻击具有广泛性。理论上说，对于所有基于 SQL 语言标准的数据库软件都是有效的，包括 MSSQL Server，Oracle，DB2，Sybase，MySQL 等。当然，各种软件有自身的特点，最终的攻击过程可能不尽相同。SQL 注入攻击的原理相对简单，易于掌握和实施，并且整个 Internet 上连接有数目惊人的数据库系统，在过去的几年里，SQL 注入攻击的数量一直在增长。

6.4.2 SQL 注入攻击实例

这里将给出一个 SQL 注入的攻击实例。结构化查询语言(Structured Query Language，SQL)是大部分关联数据库提供的一门语言，是作为应用程序与数据库进行通信的方法。

SQL 注入的原理很简单：利用未过滤的程序变量，插入任意 SQL 语句。

下面是一条典型的 SQL 语句：

```
select id, forename, surname from authors,
```

选择 authors 表中字段 id，forename，surname 的记录。

具体选择其中一条记录时可以使用这样的查询语句：

```
select id, forename, surname from authors where forename = 'john' and surname = 'smith',
```

这里值得注意的是 john 和 smith 都被包含在单引号中，如果提交：

```
forename: 'jo'hn;
surname: smith;
```

查询语句就会变为：

```
select id, forename, surname from authors where forename = 'john' and surname = 'smith',
```

当数据库执行这条语句时，会返回这样的信息：

```
Server: Msg 170, Level 15, State 1, Line 1;
```

```
Line 1: Incorrect syntax near 'hn';
```

这是由插入其中的单引号引起的,如果提交下面的输入,整个 authors 表会被删除。

```
Forename: jo'; drop table authors - ;
Surname: ;
```

从上面可以看到,在提交输入时,可以构造特殊的字符串攻击数据库。这发生在数据库的查询过程中,所以通过附加精心构造的查询语句就可以渗透到数据库中。

6.4.3 防范 SQL 注入攻击

防范 SQL 注入攻击需要遵循几条基本的规则:

(1) 在构造动态 SQL 语句时,一定要使用类安全(type-safe)的参数加码机制。大多数的数据 API,包括 ADO 和 ADO. NET,有这样的支持,即允许指定所提供的参数的确切类型(譬如,字符串,整数,日期等),可以保证这些参数被恰当地 escaped/encoded 了,来避免黑客利用它们。一定要从始至终地使用这些特性。

(2) 在部署应用前,始终要做安全审评(security review)。建立一个正式的安全过程(formal security process),在每次进行更新时,对所有的编码做审评。SQL 注入攻击从本质上看,就是攻击者利用构造的畸形 SQL 语句通过开放的 Web 服务器输入接口,把其发送到数据库服务器执行。如果 Web 服务器的接口程序没有进行细致过滤就被送往数据库服务器执行,此时非法的查询语句会取得其欲获得的多数敏感信息。通常把对输入数据审核的技术分成以下几类:对非法数据进行修整而让它变成合法数据、拒绝预知的非法语句的输入、只接受已知的合法数据。

(3) 不要把敏感性数据在数据库里以明文存放。密码应该总是在单向函数变换过后再存放。

(4) 确认已编写了自动化的单元测试,来特别校验数据访问层和应用程序不受 SQL 注入攻击。

(5) 锁定数据库的安全,只给访问数据库的 Web 应用功能所需的最低的权限。如果 Web 应用不需要访问某些表,那么确认它没有访问这些表的权限。如果 Web 应用只需要只读的权限从 account payables 表来生成报表,那么确认禁止它对此表的 insert/update/delete 的权限。

6.5 分布式拒绝服务攻击

分布式拒绝服务攻击(Distributed Denial of Service,DDoS)是一种基于 DoS 的特殊形式的拒绝服务攻击,是一种分布、协作的大规模攻击方式,主要攻击较大的站点,比如商业站点、搜索引擎等站点。

分布式拒绝服务攻击是在传统的 DoS 攻击基础之上产生的一类攻击方式。单一的 DoS 攻击一般是采用一对一方式的,当攻击目标 CPU 速度低、内存小或者网络带宽小等各项性能指标不高时,它的效果是明显的。随着计算机与网络技术的发展,计算机的处理能力

迅速增长，内存大大增加，同时也出现了千兆级别的网络，这使得攻击的困难程度加大了，目标对恶意攻击包的“消化能力”加强了不少，例如攻击软件每秒钟可以发送 3000 个攻击包，但主机与网络带宽每秒钟可以处理 10 000 个攻击包，这样攻击就不会产生明显的效果。

为达到攻击目的，产生了分布式的拒绝服务攻击方式。如果计算机与网络的处理能力加大了 10 倍，用一台主机进行 DoS 攻击不起作用的话，那么攻击者使用 100 台攻击机同时攻击，甚至更多的机器同时攻击，那么目标主机就会很快瘫痪。分布式拒绝服务攻击一般基于客户机/服务器模式，利用更多的傀儡机发起进攻，以比从前更大的规模来进攻受害者。

分布式拒绝服务攻击是指借助于客户/服务器技术，将多个计算机联合起来作为入侵平台，对一个或多个目标发动 DoS 入侵，从而成倍地提高拒绝服务攻击入侵的威力。通常，入侵者使用一个偷窃的账号将 DDoS 主控程序安装在一个特定计算机上，在一个特定的时间主控程序与大量的代理程序通信，代理程序之前已经被安装在 Internet 上的许多计算机上，代理程序收到指令时就进行入侵。利用客户/服务器技术，主控程序能在几秒内激活成百上千个代理程序的运行。

分布式拒绝服务攻击甚至只要一台主机和一个 Modem 就可实现，与 DoS 不同的是，分布式拒绝服务攻击是利用一批受控制的计算机向一台计算机发起攻击，这样来势迅猛的攻击令人难以防备，因此具有较大的破坏性。

6.5.1　分布式拒绝服务攻击原理

如图 6-7 所示，一个典型的 DDoS 攻击体系分成四大部分，攻击者、主控端、代理端与受害者。其中攻击者对主控端、代理端有控制权或者是部分的控制权，称其为傀儡机。

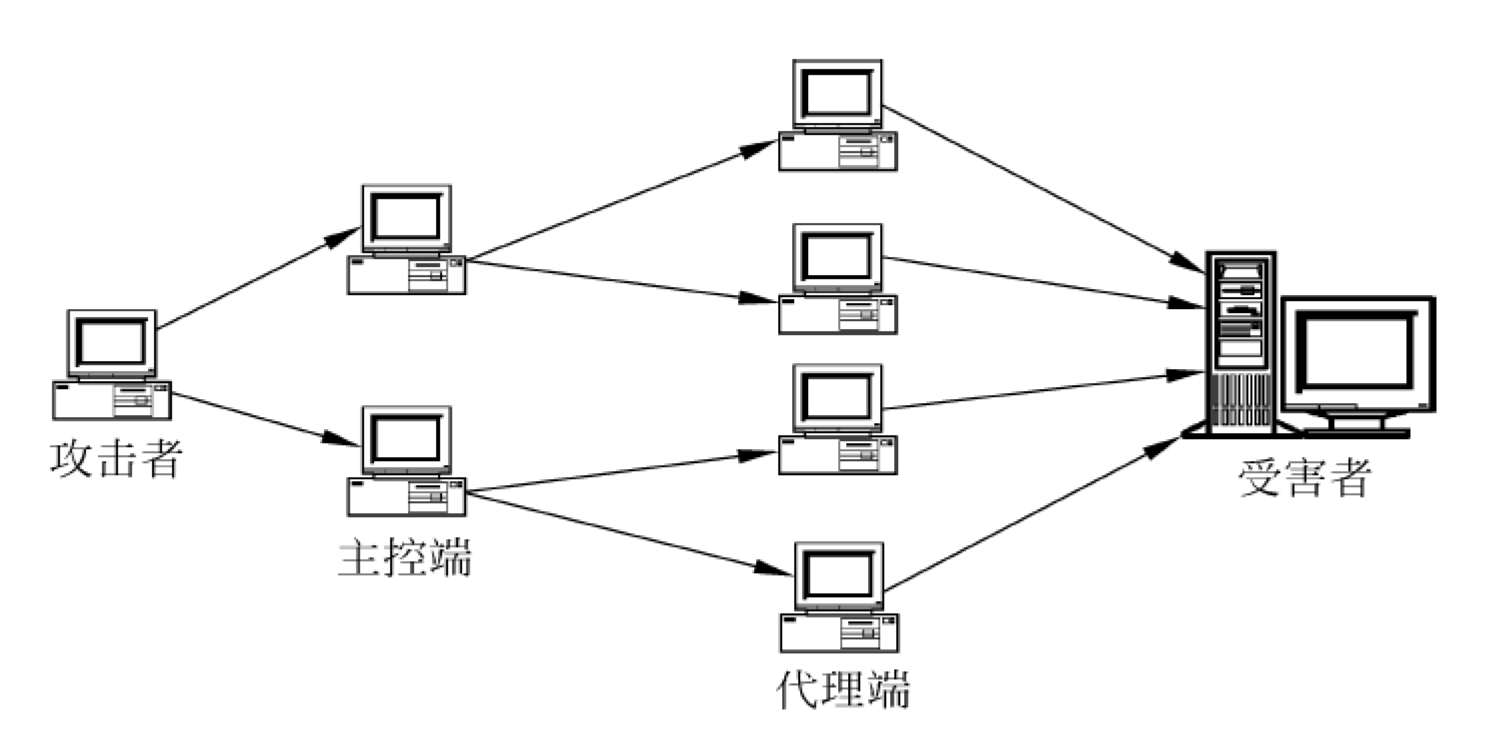


图 6-7　分布式拒绝服务攻击的原理

(1) 攻击者：攻击者所用的计算机是攻击主控台，可以是网络上的任何一台主机。攻击者操纵整个攻击过程，它向主控端发送攻击命令。通常它会利用各种手段隐藏自己不被别人发现。

(2) 主控端：主控端是攻击者非法侵入并控制的一些主机，这些主机还分别控制大量的代理主机。主控端主机的上面安装了特定的程序，因此它们可以接受攻击者发来的特殊指令，并且可以把这些命令发送到代理主机上。

(3) 代理端：代理端也是攻击者侵入并控制的一批主机，它们运行攻击器程序，接受和运行主控端发来的命令。代理端主机是攻击的执行者，真正向受害者主机发送攻击。

发动 DDoS 攻击一般分为两个阶段。

（1）初始的入侵阶段：攻击者首先扫描远程脆弱主机和服务器，并采用黑客入侵手段控制大量的主机和服务器，作为主控端和攻击端。

（2）攻击阶段：攻击者向主控端发出攻击命令，主控端再发送给每个攻击端，最后由攻击端向被攻击的目标发送 DoS 攻击数据包，对目标主机进行大规模拒绝服务攻击。

6.5.2 DDoS 攻击方式及分类

按照攻击路径可以将 DDoS 分为两大类：直接攻击和反射攻击(Distributed Reflection Denial of Service，DRDoS)。在直接攻击当中，攻击者直接向受害者发送大量的攻击包，数据包的类型可以是 TCP、ICMP、UDP，或是它们的混合。在反射攻击中，代理端的源 IP 地址是受害端的 IP 地址。常见的反射攻击方式有 Smurf、SYN 洪泛、RST 洪泛、ICMP 洪泛、DNS 响应洪泛等。当前常见攻击方式有四种：SYN Flooding、UDP Flooding、ICMP Flooding、Smurf。其中 SNY Flooding、UDP Flooding 和 ICMP Flooding 属于直接攻击，而 Smurf 属于反射攻击。

1. SYN Flooding 攻击

SYN Flooding 是利用了传输层的 TCP 协议的弱点。我们知道 TCP 是基于连接的，也就是说在数据传输之前，用 TCP 协议进行传输的双方需通过“三次握手”来完成一个连接。针对每个连接，连接双方都要为该连接分配内存资源：Socket 结构、Internet 协议控制块结构、TCP 控制块结构等。

在建立连接的三次握手中假设客户端向服务器端发送了 SNY 报文后突然死机或掉线，那么服务器端在发出 SYN + ACK 应答报文后是无法收到客户端的 ACK 报文的，在这种情况下服务器一般会重试，再次发送 SYN + ACK 应答报文并等待一段时间后丢弃这个未完成的连接，这段时间的长度称为 SYN Timeout。一个用户出现异常导致服务器端的一个线程等待 1 个 SYN Timeout 并没有什么问题，但是如果这段时间内有大量的类似情况的 SNY 数据包的话，服务器端将要维护一个规划庞大的半连接列表，这会消耗非常多的系统资源。

在 SYN Flooding 攻击中，攻击者首先向攻击目标发出伪装源 IP 地址的 SNY 包，也就是连接请求包，伪装的源 IP 地址是一个虚假的或者是一个无法到达的 IP 地址。因为如果该源 IP 地址是可达的，那么拥有这个 IP 地址的主机收到被攻击者的 SYN+ACK 包后，会回复 RST 包，即连接重置，被攻击者则会删除该连接，这样攻击者就无法达到目的了。如果源 IP 地址是伪造的，当被攻击者收到连接请求后，被攻击者回复 SYN+ACK 包，并分配资源给该连接。由于 SYN+ACK 包是发给伪造 IP 地址的主机，因此不会有任何响应。于是被攻击者继续发送 SYN+AKC 包，并将该连接放入端口的队列中。一般操作系统都有默认的回复次数和超时时间，在 Windows 2000 操作系统中会重复发送 SYN + AKC 包 5 次，每次重发的等待时间增加一倍：即第一次等待 3 秒，第二次等待 6 秒，第三次等待 12 秒，第四次等待 24 秒，到第五次等待 48 秒。若还收不到回应，还需等待 96 秒，被攻击者才会释放划分给该连接的资源。在资源被释放前，已经浪费了系统 189 秒的时间。由于端口队列的大小是有限的，当处于半连接状态一段时间后会被清除，但如果不断的接收到大量的伪装源 IP 的连接请求包，将抵消清除的效果，最终使被攻击者的资源耗尽，无力响应合法请求。

2. UDP Flooding 攻击

用户数据报协议(User Datagram Protocol,UDP)是 ISO 参考模型中一种无连接的传输层协议,提供面向事务的简单不可靠信息传送服务。它服务于很多应用层协议,包括 NFS、简单网络管理协议(Simple Network Management Protocol,SNMP)、DNS 以及简单文件传输系统(Trivial File Transfer Protocol,TFTP)等。

因 UDP 位于传输层需要提供进程通信能力,因此提出了协议端口的概念,用于标识通信的进程。应用程序通过系统调用与某(些)端口建立连接后,传输层传给该端口的数据都被相应进程接收。UDP 有一些保留端口,同一个保留端口在 UDP 中可能对应于不同类型的应用进程,也可能对应于相同类型的服务进程。一个 UDP 端口是一个可读、可写的软件结构,内部有一个接收报文缓冲区。发送数据时 UDP 软件构造一个数据报,然后将它交给 IP 软件,便完成所有的工作。接收数据时,UDP 软件先要判断接收数据报的端口是否与当前使用的某端口匹配。如果匹配,则将数据报放入相应接收队列,否则,抛弃该数据报,并向信源端发送"端口不可到达"的 ICMP 报文。虽然端口匹配成功,但如果相应端口队列已满,UDP 也会抛弃数据报。

如用 Chargen(字符产生器)和 Echo 服务传送毫无用处的数据来占用所有的带宽。Echo 如果收到一个包,就会把包中的负载按原样发回。如果向 Chargen 端口送一个任意字符,它将返回一个假的随机字符串。通过伪造与某一主机的 Chargen 服务之间的一次 UDP 连接,回复地址指向开着 Echo 服务的一台计算机,这样就生成在两台计算机之间的足够多的无用数据流,如果有过多的无用数据流就会导致带宽的拒绝服务攻击。

一个攻击者假冒某台计算机的 Echo 端口向不同的计算机发起多个 Chargen 连接,其目的就是在被假冒主机和这些主机间产生 Echo、Chargen 循环往返效果,使这些主机充满这些无用的数据包,浪费它们的带宽和 CPU 资源。另外成百上千台代理端同时向被攻击计算机长时间、连续、大量地发送 UDP 数据包,使得被攻击计算机耗费大量的 CUP 资源处理,疲于奔命,最终使被攻击计算机瘫痪。

3. ICMP 攻击

ICMP(Internet Control Message Protocol,控制消息协议),是 TCP/IP 协议族的子协议,用于在 IP 主机、路由器之间传输控制消息。控制消息是指网络是否畅通、主机是否可达、路由是否可用等网络消息。这些控制消息虽然并不传输用户数据,但是对于用户数据的传递起着重要的作用。

ICMP 协议对于网络安全具有重要的意义。ICMP 协议本身的特点决定了它非常容易被用于攻击网络上的路由器和主机。

ICMP Flooding 攻击就是成百上千台代理端同时向目标主机长时间、连续、大量地发送 ICMP 数据包,最终使系统瘫痪。大量的 ICMP 数据包会形成"ICMP 风暴",使得目标主机耗费大量的 CPU 资源处理,其方式与 UDP Flooding 攻击有些类似,只不过是发送的 ICMP 数据包。现在一些路由器采取以下策略来防止 ICMP Flooding:在广域网接口和局域网接口上,当路由器的接口接收到每秒超过 1000 个 ICMP 数据包时,路由器将临时关闭这个 IP 地址的所有数据进入。

4. Smurf 攻击

Smurf 攻击是根据最初能发起这种攻击的攻击程序命名的，Smurf 是一种间接、借力的攻击方式。它是一种利用了 ICMP 协议的反射攻击方式。所谓的 Smurf 攻击是指，攻击者在远程机器上发送 ICMP 应答请求服务，其目标主机不是某一个主机的 IP 地址，而是某个网络的广播地址，其请求包的源 IP 不是发起攻击的 IP 地址，而是加以伪装的将要攻击的目标主机的 IP 地址。大量主机收到 ICMP 应答请求服务包后，按照源 IP 返回请求信息，从而导致受攻击主机的服务性能下降，甚至崩溃。

任何连接到互联网上的支持 ICMP 请求响应的网络设备都可能成为这种攻击的目标。设想发送一个 IP 包到广播地址 192.168.1.0，假设这个网络中有 50 台计算机，那么将会收到 50 次的应答，广播地址在这里起到了放大器的作用。Smurf 攻击就是利用了这种作用，如果 A 发送 1KB 大小的 ICMP Echo Request 到广播地址，那么 A 将收到 IKB * N 的 ICMP Echo Reply 应答数据，其中 N 为网络中计算机的总数。当 N 等于 100 万时，产生的应答将达到 1GB，这将会大量消耗网络资源。如果 B 假冒 A 的 IP 地址向广播地址发送 ICMP Echo Request，那么收到应答的是 A，对 A 来说就形成了一次 DDoS 攻击。

Smurf 攻击行为的完成涉及三个角色：攻击者、中间脆弱网和目标受害者。攻击者伪造一个 ICMP Echo 请求包，包中的源 IP 地址为目标受害者的 IP 地址，目的 IP 地址为中间脆弱网络的广播地址，并将该请求包发送到中间脆弱网。中间脆弱网中的主机收到这个 ICMP Echo 请求包时，会以 Echo 响应包作为应答，应答包被发送到目标受害者。这样大量同时返回的 Echo 响应数据包会造成目标主机所在的网络严重拥塞，丢包，甚至完全不可用等现象，从而达到攻击目的。尽管中间脆弱网络没有被称为受害者，但实际上中间网络的性能也会遭受严重影响。

常用的 DDoS 工具有 Trinoo、TFN、Stacheluraht、TFN2K 和 Trinity v3 等。

6.6 TCP 会话劫持

会话劫持(session hijack)，就是在一次正常的会话过程当中，攻击者作为第三方参与到其中，他可以在正常数据包中插入恶意数据，也可以在双方的会话当中进行监听，甚至可以是代替某一方主机接管会话。可以把会话劫持攻击分为两种形式：被动劫持与主动劫持；被动劫持实际上就是嗅探双方会话的数据，获得敏感数据；而主动劫持则是将会话当中的某一台主机踢下线，然后由攻击者取代并接管会话。

1. TCP 会话劫持的原理

TCP 协议通过三次握手建立连接：①客户机发送一个 SYN 报文，请求连接服务器的某个端口，报文里面带有初始序列号 a；②服务器返回包含初始序列号 b 的 SYN+ACK 报文，确认号设置为 a+1；③客户发送序列号 a+1，确认号 b+1 的 ACK 报文。

如果在三次握手过程中序列号及确认号都正确，客户机便认为与服务器建立了 TCP 连接。TCP 会话劫持就是在一次正常的通信过程中，有第三方参与到其中，在基于 TCP 的会话里发出更多的信息，即从会话双方直接联系变成与第三方联系。

会话劫持攻击可分为两种类型：①中间人攻击(Man In The Middle，MITM)；②注射式攻击(injection)。

下面简要介绍这两种攻击类型。

1) MITM 攻击

要想正确地实施中间人攻击，攻击者首先需要使用 ARP 欺骗或 DNS 欺骗，将会话双方的通讯流暗中改变，而这种改变对于会话双方来说是完全透明的。

关于 ARP(Address Resolution Protocol，地址解析协议)欺骗的原理：在交换式网络中，交换机在处理数据的时候，它会根据自己机器内部的一个 MAC 到端口的数据表来查询符合要求的 MAC 地址数据包该发往哪个端口。这张表从交换机开机的时候就存在，在每个端口第一次数据传送的时候就会记录对应的端口的 MAC 地址。而通过发送伪造的 MAC 地址数据包到交换机，就可以欺骗交换机刷新自己的 MAC 地址到端口的数据表。如 A 主机连接在 2 口，攻击者在 4 口，要监听 A 主机的数据，那么就需要伪造一个 ARP 数据包，告诉交换机 A 主机 MAC 地址是在 4 口，那么交换机就会将本来发送到 A 主机的数据会转送到 4 口上，这个时候攻击者就可以监听传送到 A 主机的数据，这个就是基于交换网络的 ARP 欺骗。

对于 DNS(Domain Name System，域名服务器)欺骗的原理是：正常的 DNS 请求，是在浏览器输入网址后，系统先查看 Hosts 文件，如果有相对应的 IP，就使用这个 IP 地址访问网站；如果没有，就去请求 DNS 服务器；DNS 服务器在接收到请求之后，解析出其对应的 IP 地址，返回给本地，最后你就可以登录网站了。而 DNS 欺骗则是，目标机将其 DNS 请求发送到攻击者那里，然后攻击者伪造 DNS 响应，将正确的 IP 地址替换为其他 IP(其中含有恶意网页)，当你登录到攻击者指定的 IP 后，不知不觉就被攻击了。DNS 欺骗也可以在广域网中进行，比较常见的有“Web 服务器重定向”、“邮件服务器重定向”等。

但是不管是 ARP 欺骗，还是 DNS 欺骗，中间人攻击都改变正常的通信流，它就相当于会话双方之间的一个透明代理，可以得到一切想知道的信息，甚至是利用一些有缺陷的加密协议来实现。

2) 注射式攻击

这是一种比中间人攻击实现起来简单些的攻击，它不会改变会话双方的通信流，而是在双方正常的通信流中插入恶意数据。在注射式攻击中，需要实现两种技术：①IP 欺骗；②预测 TCP 序列号。

对于 IP 欺骗，有两种情况需要用到：①隐藏自己的 IP 地址；②利用两台机器之间的信任关系实施入侵。对于基于 TCP 协议的注射式会话劫持，攻击者应先采用嗅探技术对目标进行监听，然后从监听到的信息中构造出正确的序列号，而猜测会话双方的序列号是非常重要的。

2. TCP 会话劫持的防范

(1) 首先应该使用交换式网络替代共享式网络，虽然像 Hunt 这样的工具可以在交换环境中实现会话劫持，但还是应该使用交换式网络替代共享式网络，因为这样可以加强防范最基本的嗅探攻击。

(2) 最根本的解决办法是采用加密通信，使用 SSH、SSL 对通信进行加密，这样会话劫持就无用武之地了。

(3) 监视网络流量，如发现网络中出现大量的 ACK 包，则有可能已被进行了会话劫持

攻击。

3. TCP 会话劫持应用

这里将介绍一种基于 TCP 会话劫持的告警方法。

当用户发现自己的网络出现问题时，都有使用打开网页的习惯去测试自己的网络是否和外界保持畅通。所以当用户被隔离时，可以利用这类习惯来进行及时通知。响应模块采取响应时，会保存被隔离用户的相关信息，数据采集模块可以读取表中的 IP 地址信息，建立一张黑名单。当黑名单内的 IP 开始和外界建立 TCP 连接时，控制器采取措施来中断 TCP 连接，并对所有 HTTP 请求返回响应伪造报文，使其重定向到安全网站提示用户当前所处的状态，以便其进行修复。

在监听到问题主机发出 SYN 报文时，如果发现这个报文的目标端口是 80 端口，则立即伪装成服务器发送 SYN＋ACK 报文给问题主机。假设问题主机发出的 SYN 报文的序列号为 a，则伪造的 SYN＋ACK 报文的序列号为 b，确认号为 a＋1。因为系统是在网络各汇聚层出口处捕获数据，而客户主机会经常试图访问外网，所以可以较好地保证伪造的报文会比真正的服务器发出的包先到达问题主机，会先处理发送的伪造报文。当其认为这是服务器发来的第二次握手报文，会发送第三次握手报文，也就是一个 ACK 报文来建立这个 TCP 连接。此时问题主机认为已经和告警系统建立了一个 TCP 连接，但其不知道自己建立了伪连接。在 TCP 连接建立以后，问题主机的浏览器会向服务器发送一个 HTTP GET 请求报文。告警子系统又先于真正的服务器收到了这个报文，将向该主机发出一个重定向 HTTP 报文，使其转到指定的安全告警页面。

习 题 6

1. 计算机病毒具有哪些特征？
2. 蠕虫和病毒有哪些不同？
3. 说明一下木马的工作原理。
4. 你知道哪些网络嗅探软件？试对其中一个进行说明。
5. 缓冲区溢出的工作原理是什么？
6. 什么是 SQL 注入攻击？
7. 分布式拒绝服务攻击的工具有哪些？它们是如何工作的？
8. 怎样防止 TCP 会话劫持攻击？

第 7 章　防　火　墙

7.1　防火墙概述

在早期的欧洲建筑，为了防止火灾的发生和蔓延，人们在房屋的周围修建矮墙作为一种防护措施，这种建筑物被称为防火墙。在计算机网络中，人们借助这个概念在内外网络的边界上建立的一种过滤和封锁系统，控制内外网的通信，称为防火墙系统，如图 7-1所示。

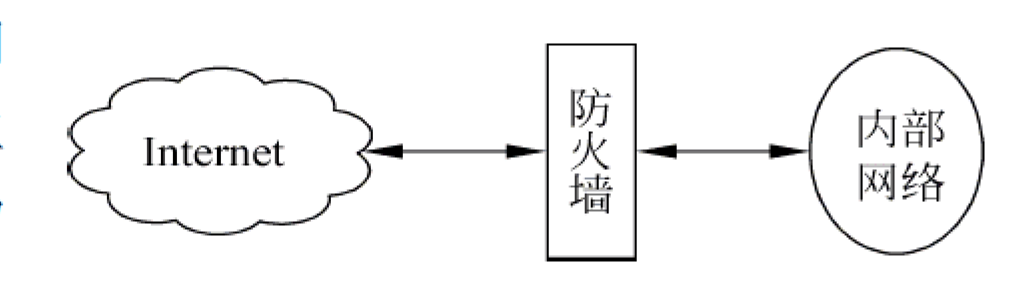


图 7-1　防火墙系统

防火墙是一种网络访问控制设备，位于两个(或多个)网络设备之间，通过执行访问控制策略来达到网络安全的目的。它隔离了内部和外部网络，是内部和外部网络通信的唯一途径，能够根据制定的访问规则对流经它的信息进行监控和审查，以保护内部网络不受攻击和非法访问。

防火墙是不同网络之间的信息的唯一入口，能根据用户的需求制定安全规则，控制出入网络的信息流，尽可能的屏蔽内部网络的拓扑结构和运行状况，保护内网的安全，通常防火墙的功能有以下这些。

1）防火墙作为网络的集中监视点，是网络安全的屏障

一个防火墙(作为阻塞点、控制点)能极大地提高一个内部网络的安全性，并通过过滤不安全的服务而降低风险。由于只有经过精心选择的应用协议才能通过防火墙，所以网络环境变得更安全。防火墙同时可以保护网络免受基于路由的攻击，如 IP 选项中的源路由攻击和 ICMP 重定向中的重定向路径。

2）隔离内外网络、保护内部网络，防止内部信息的外泄

通过利用防火墙对内部网络的划分，可实现内部网重点网段的隔离，从而限制局部重点或敏感网络安全问题对全局网络造成的影响。再者，隐私是内部网络非常关心的问题，一个内部网络中不引人注意的细节可能包含了有关安全的线索而引起外部攻击者的兴趣，甚至因此而暴露了内部网络的某些安全漏洞。使用防火墙就可以隐蔽那些透露内部细节的服务，如 Finger、DNS 等服务。Finger 显示了主机的所有用户的注册名、真名，最后登录时间和使用 shell 类型等。Finger 显示的信息非常容易被攻击者所获悉，从而可以知道一个系统使用的频繁程度，这个系统是否有用户正在连线上网，这个系统是否在被攻击时引起注意等。防火墙同样可以阻塞有关内部网络中的 DNS 信息，这样一台主机的域名和 IP 地址就不会被外界所了解。

3）强化网络安全策略

通过以防火墙为中心的安全方案配置，能将所有安全软件(如口令、加密、身份认证、审计等)配置在防火墙上。与将网络安全问题分散到各个主机上相比，防火墙的集中安全管理

更经济。例如在网络访问时，一次一密口令系统和其他的身份认证系统完全可以不必分散在各个主机上，而集中在防火墙上。

4）有效记录和审计内外网络活动

如果所有的访问都经过防火墙，那么，防火墙就能记录下这些访问并做日志记录，同时也能提供网络使用情况的统计数据。当发生可疑动作时，防火墙能进行适当的报警，并提供网络是否受到监测和攻击的详细信息。另外，收集一个网络的使用和误用情况也是非常重要的。首先的理由是可以清楚防火墙是否能够抵挡攻击者的探测和攻击，并且清楚防火墙的控制是否充足。而网络使用的统计数据对网络需求分析和威胁分析等而言也是非常重要的。

7.2 防火墙的基本技术

防火墙技术可根据防范的方式和侧重点的不同而分为很多种类型，但总体来讲可分为包过滤、应用级网关和代理服务器（电路网关）等几大类型。

7.2.1 包过滤

包过滤，简单地说就是一个根据数据包头部信息来选择允许或拒绝数据包在网络中传输的过程。包过滤技术应用在网络层，监视网络上流入流出的数据包，过滤那些可疑的包。

一般将包过滤功能部署在路由器上，路由器中包含过滤器（或过滤软件）。过滤器选择数据包的依据是系统内设置的过滤规则——访问控制表（Access Control List，ACL）。表中的规则都是基于数据包头的包头信息制定的。通过检查数据流中每一个数据包的源地址、目的地址、端口号、协议状态等，判断是否允许数据包通过，如图 7-2 所示。

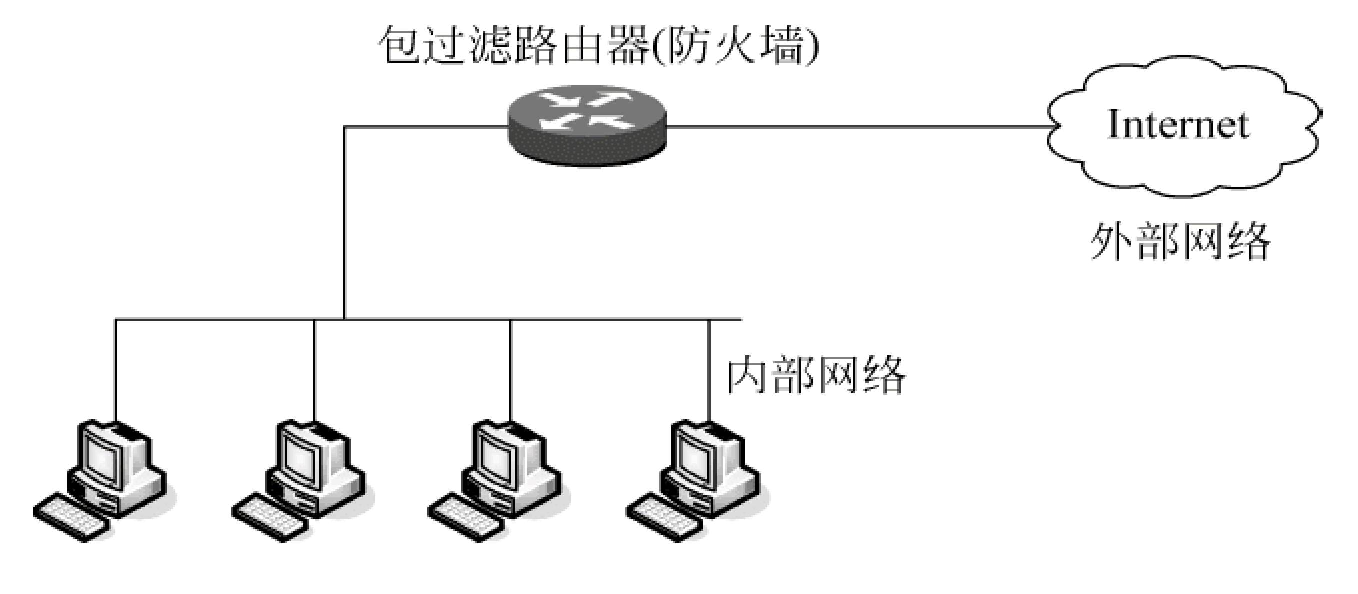


图 7-2　包过滤路由器

1. 包的构造与解析

图 7-3 所示的是数据包的构造和它在 TCP/IP 协议各层上的操作，分别是应用层、传输层、网络层和网络接口层。

数据包的构造有点像洋葱，它是由各层连接的协议组成的。在每一层，数据包都由首部和数据两部分组成。在首部存放与这一层相关的协议信息，其他为这一层的数据信息，这些数据信息包含了上一层的全部信息。数据包的构造过程就是从上一层获取的信息加上本层的首部信息，这就是数据包的封装过程，如图 7-3 所示的实线箭头所指的路径。与数据包封

装过程相反，在网络的接收端的工作是包的解析过程，将收到的收据包按从下至上的顺序去掉首部信息，如图 7-3 所示的虚线箭头所指的路径。

对于包过滤系统来说，包的最重要信息是各层依次加上的首部信息，而包过滤工作在网络层，即对 IP 数据包进行分析。

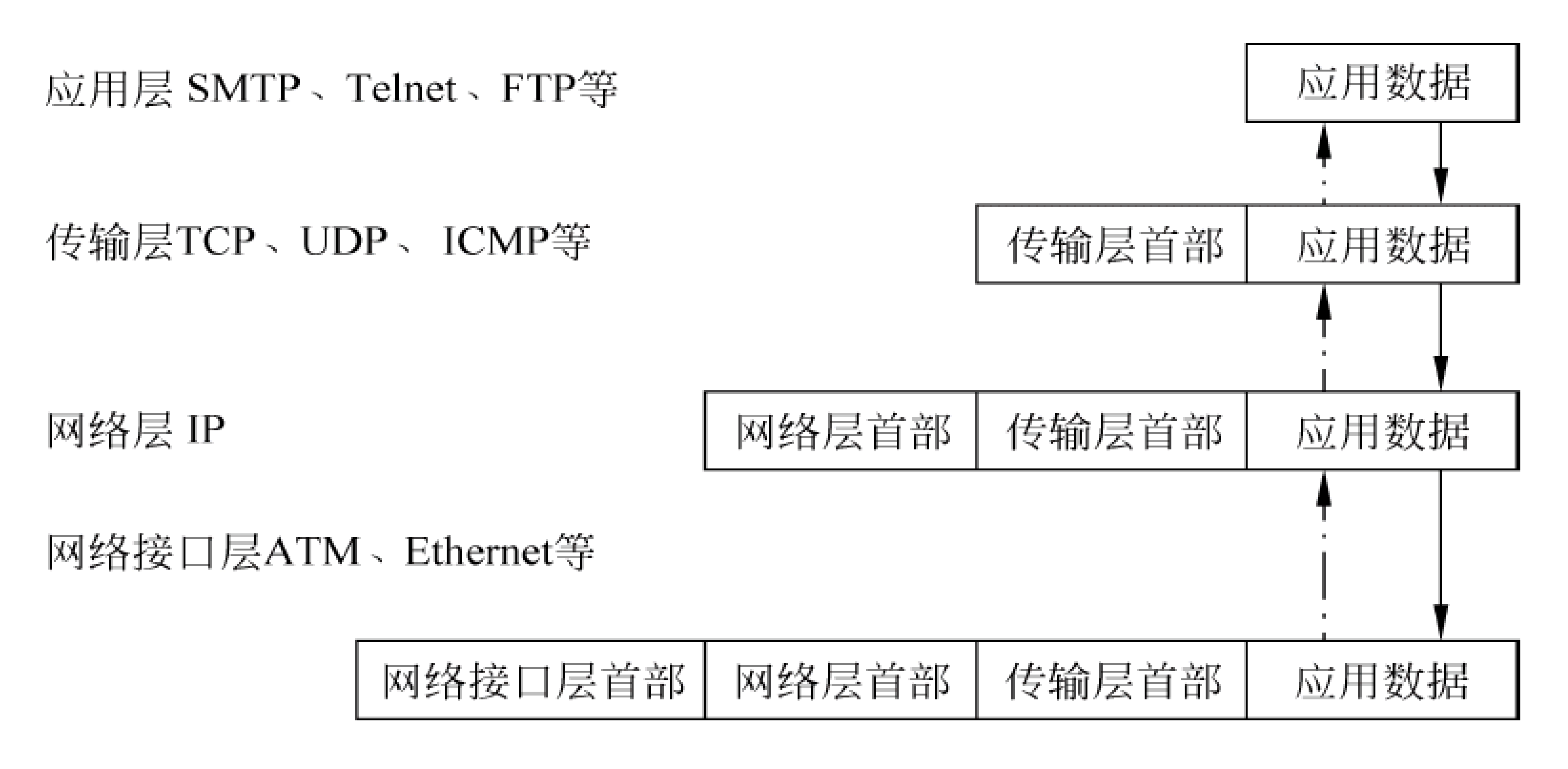


图 7-3　数据包的封装

在 TCP/IP 网络上通信的主机，在发送端将数据进行分组，封装为 IP 数据包，包中包含发送者的 IP 地址和接收者的 IP 地址信息。当这些数据包被送上互联网络时，路由器会读取接收者的 IP 并选择一条合适的物理线路发送出去，信息包可能经由不同的路线到达目的地，当所有的包抵达目的地后会重新组装还原。

包过滤防火墙会检查所有通过的信息包中的 IP 地址，并按照系统管理员所给定的过滤规则进行过滤。如果对防火墙设定某一 IP 地址的站点为拒绝访问，从这个地址来的所有信息都会被防火墙屏蔽掉。包过滤技术是防火墙为系统提供安全保障的主要技术，它通过设备对进出网络的数据流进行有选择的控制与操作，包过滤操作通常在选择路由的同时对数据包进行过滤(通常是对从互联网络到内部网络的包进行过滤)。用户可以设定一系列的规则，指定允许哪些类型的数据包可以流入或流出内部网络；哪些类型的数据包的传输应该被拦截。

基于包过滤技术的防火墙包括静态包过滤防火墙和动态包过滤防火墙。

2. 静态包过滤

这类防火墙几乎是与路由器同时产生的，它是根据定义好的过滤规则审查每个数据包，以便确定其是否与某一条包过滤规则匹配，所以称之为静态包过滤。过滤规则基于数据包的报头信息进行制订。报头信息中包括 IP 源地址、IP 目标地址、传输协议(TCP、UDP、ICMP 等)、TCP/UDP 目标端口、ICMP 消息类型等。通常的判断依据(IP 包)包括：数据包协议的类型(TCP、UDP、ICMP、IGMP)；数据包的源、目的 IP 地址；目的端口、源端口(FTP 、HTTP、DNS)；TCP 选项标志(SYN 、ACK、FIN、RST)；数据包的流向。

包过滤防火墙逐一审查到达的 IP 数据包以判定它是否与其他包过滤规则相匹配。每个包有两个部分：数据部分和包头。过滤规则以用于 IP 顺行处理的包头信息为基础，不理会包内的正文信息内容。根据制定的规则，如果找到一个匹配，且规则允许/拒绝该数据包，则发送或拒绝该数据包；如果与设定的所有规则都不匹配，则防火墙采用默认的策略(通常默认的规则有两种：无匹配即拒绝数据包或无匹配即允许数据包)。

例如下面是包过滤规则设定的例子。* 表示匹配所有；假设默认的规则是拒绝，内部主机的 IP 地址为 192.168.0.2。该规则允许向外发送 Web 请求，且允许 Web 站点的响应包进入。

规　　则	源　地　址	源　端　口	目的地址	目标端口	流　　向
接收	192.168.0.2	*	*	80443	流出
接收	*	80443	192.168.0.2	1024—65535	流入

而在下面的例子中，所有来自外部主机的数据包都将被阻止。

规　　则	源　地　址	源　端　口	目的地址	目标端口	流　　向
拒绝	*	*	192.168.0.2	1024—65535	流入

静态包过滤有明显的特点：不用改动应用程序、一个包过滤路由器能协助保护整个网络、过滤路由器速度快、效率高。包过滤路由器对终端用户和应用程序是透明的。当数据包过滤路由器决定让数据包通过时，它与普通路由器没什么区别，甚至用户没有认识到它的存在，因此不需要专门的用户培训或在每台主机上设置特别的软件。

同时静态包过滤的缺陷也很明显：为完成一个特定任务，包过滤的规则定义可能比较复杂，而且不容易验证逻辑的正确性；用包过滤控制 RPC、X-Window 和 FTP 比较困难，这些协议通常分配的端口号不固定，因此包过滤路由器很难对其进行有效控制；且不能防止地址欺骗，IP 地址伪造很容易；正常的数据包过滤路由器无法执行某些安全策略。

3. 动态包过滤

静态包过滤对 IP 数据包的端口信息作为一项判断依据，这样当遇到利用动态端口的协议时就会出现问题。例如，在 FTP 协议中，事先无法知道哪些端口需要打开，而如果采取静态包过滤的方法，又希望使用 FTP 服务，就需要将可能用到的端口全部打开，这是个很大的端口范围，很可能被攻击者利用。

动态包过滤，也称状态检测技术，采用动态设置包过滤规则的方法。采用这种技术的防火墙对通过其建立的每一个连接都进行跟踪，并且根据需要可动态地在过滤规则中增加或更新条目。动态包过滤防火墙，检查数据包的头部信息，来判断此连接是否需要打开某个端口。当数据包传输完毕，到达目的地址时，端口又马上恢复为关闭状态。动态端口过滤技术使得暂时性的端口只有在需要的时候才会被打开，并且会拒绝所有永久性占用端口的请求。

动态包过滤防火墙的过滤规则根据会话状态的不同而动态改变，并且防火墙采用了一个执行安全策略的状态检测模块。当通过防火墙建立从内部主机访问外部主机的 TCP/UDP 会话时，防火墙会将该会话的信息记录在一张临时的状态会话表中，并且利用该状态表跟踪每一个会话状态。同时，防火墙还会建立一些响应的过滤规则。所记录的会话状态信息包括：源/目的 IP 地址、端口号等，对于 TCP 会话，还应包括序列号信息以及与某一主机相关的 TCP 连接的附加标志位信息。在对数据包进行过滤时，不仅要依据规则表，还要考虑数据包状态是否符合会话所处的状态。只有当数据包的状态与当前会话的状态信息一致时，才被允许通过。状态会话信息是临时的，当会话结束时，有关该会话的信息以及相应的过滤规则就被删掉。

7.2.2　应用层代理

代理技术是与包过滤不同的一种防火墙技术。包过滤主要根据特定的过滤条件来判定是否允许数据包通过，一旦条件满足，防火墙的内外两侧的主机将建立直接连接，数据就可以传输了。这样外部的主机就有机会连接内部网络结构，威胁内部网络安全，因此引入了代理技术。

代理技术的基本思想就是在两个网络之间设置一个“中间检查站”，两边的网络应用可以通过这个检查站互相通信，但是它们不能建立直接的连接、越过检查站直接通信。这个中间检查站就是代理服务器，它运行于两个网络之间，对网络之间的请求进行检查，并对合法信息进行转发，如图 7-4 所示。通常代理技术分为：应用层代理（应用网关）、电路网关。

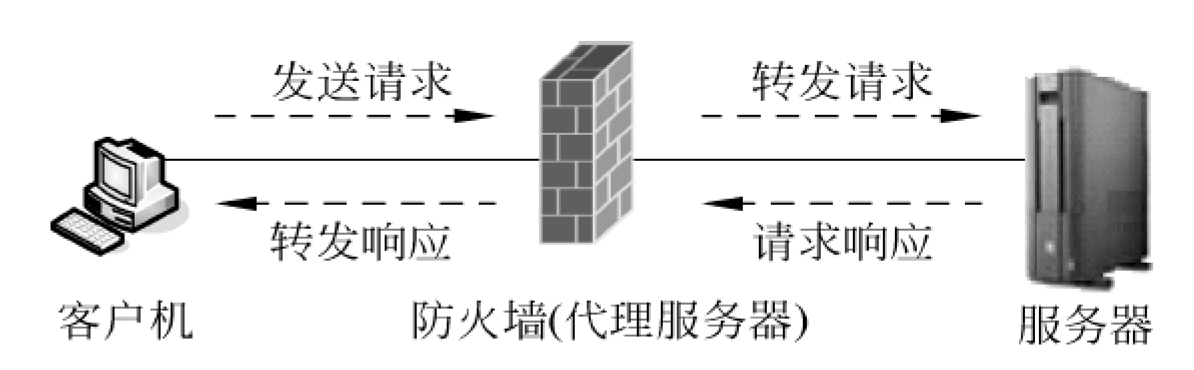


图 7-4　代理服务器工作过程

应用层代理也称应用网关，就是通常提到的代理服务器，是指运行在防火墙主机上的特殊应用程序或者服务器程序，这些程序根据安全策略接受用户对网络的请求，并在用户访问应用信息时依据预先设定的应用层协议安全规则进行信息过滤。

应用层代理是在网络应用层通过建立协议过滤和转发功能实现的，工作于应用层（如图 7-5 所示），适用于特定的互联网服务，如超文本传输（HTTP）、远程文件传输（FTP）等。它针对特定的网络应用服务协议使用制定的数据过滤逻辑，并在过滤的同时对数据包进行必要的分析、登记和统计，形成报告。

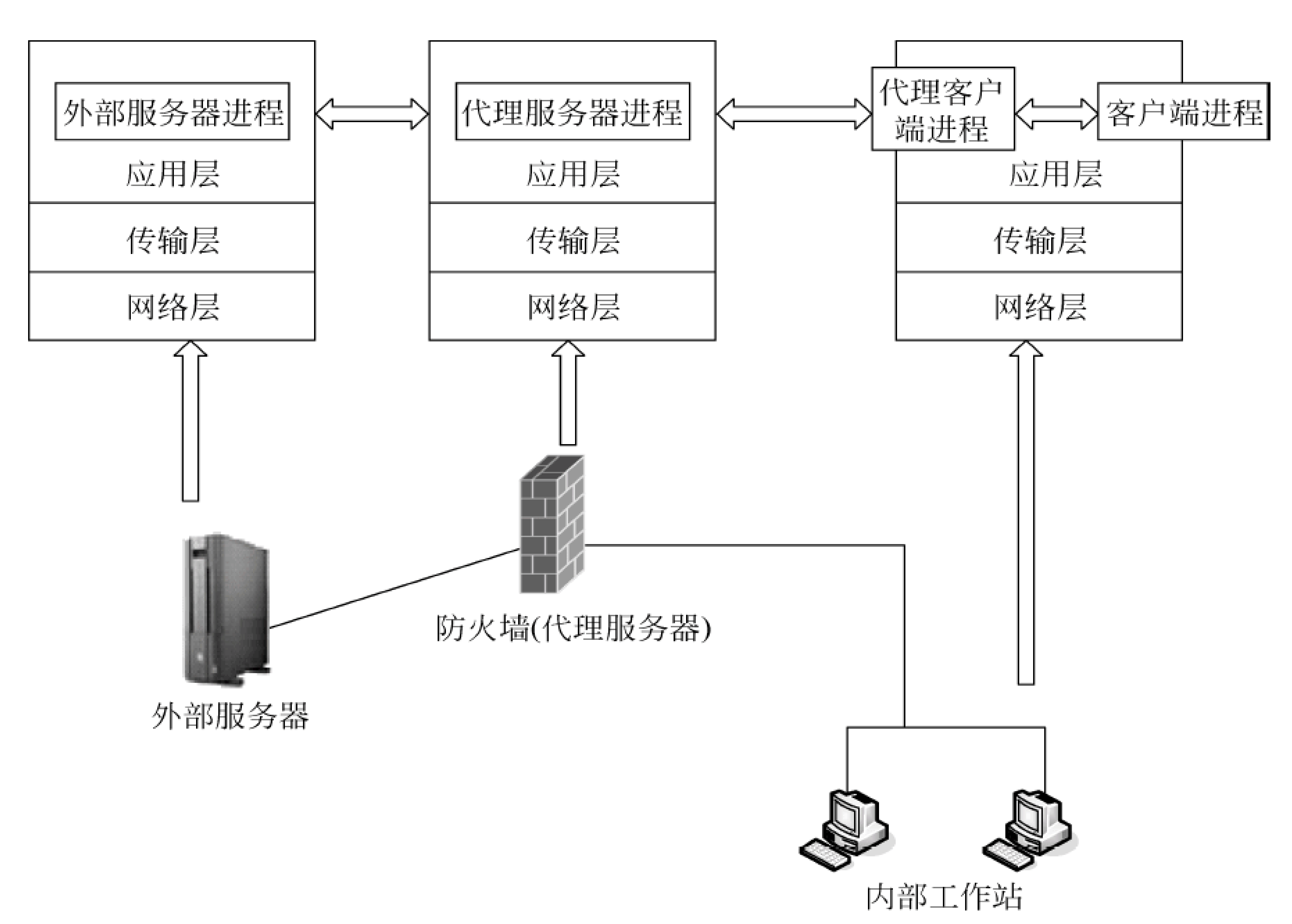


图 7-5　应用层代理

使用代理服务器后,内网的用户只能将应用层协议请求提交给代理,由代理访问请求的网络资源,并将结果返回给用户。因此,内部网络用户与外部网络资源之间不建立直接的网络连接或者直接的网络通信,所有信息交互必须借助于应用层代理的中继功能。实际上,内部网络用户与应用层代理之间建立应用层连接,而应用层代理与外部网络资源之间也将建立应用层连接。

应用层代理防火墙介于内部网络与外部网络资源之间,它对于客户来说像是一台真的服务器,而对外界的服务器来说,它又是一台客户机;当代理服务器接收到用户对某站点的访问请求后会检查该请求是否符合规定,如果规则允许用户访问该站点,代理服务器会像一个客户一样去那个站点取回所需信息再转发给客户。代理服务器通常都拥有一个高速缓存,这个缓存存储有用户经常访问的站点内容,在下一个用户要访问同一站点时,服务器就不用重复地获取相同的内容,直接将缓存内容发出即可,既节约了时间也节约了网络资源。当一个远程的用户请求内部服务时,它首先与这个代理相连,经过鉴别后,再由代理连接到目的主机,同时将服务器的响应传送给所代理的客户。

应用层代理由代理服务器和代理客户端两个部件组成,只有通过两个部件的协同工作才能实现代理功能。代理服务器是运行在防火墙上的应用程序,而代理客户端是对普通客户程序进行修改后的特别版本,代理客户端与代理服务器建立应用层连接而不与真正的外部资源服务器建立连接。

应用层代理的优点如下:

(1) 应用层代理介于内部用户和外部网络之间,不允许内、外网用户直接相连,从外部只能看到该代理服务器而无法获知任何的内部资源,如用户的 IP 地址等,内部网络的安全性比较高;

(2) 应用层代理比单一的包过滤更为可靠,而且会详细地记录访问信息和日志记录,例如,在一个 Http 连接中,包过滤只能记录单个的数据包,而应用层代理还可以记录文件名、URL 等;

(3) 应用层代理支持对单个用户授权、提供用户认证等安全策略。

但是应用层代理也存在不足之处。首先,因为代理相当于一个简化的服务器/客户端,且每个合法通过的服务都要在代理上启动服务器和客户端。使得网络传输效率比较低,访问速度变慢。其次,代理需要特定的客户端程序,且通常只能针对特定的应用,用户不能使用未被支持的服务。

7.2.3 电路网关

与应用网关相类似,另一种类型的代理服务器接收客户端的连接请求,代表客户端完成网络连接。建立起一个回路,对数据包的转发起作用,并把数据包提交给网络应用层处理,故被称为电路网关。电路网关是建立在传输层上的一种代理方法,它不像应用网关针对专门的应用层协议,是在内、外网络主机之间建立一个虚拟电路进行通信,相当于在防火墙上直接开了个口子进行传输,不像应用网关那样能严密地控制应用层的信息。

电路层网关只依赖于 TCP 连接,并不进行任何附加的包处理或过滤。电路层网关就像电线一样,只是在内部连接和外部连接之间来回复制字节。但由于连接似乎是起源于防火墙,从而隐藏了受保护网络的有关信息。连接的发起方不直接与响应方建立连接,而是与电

路网关进行交互，由电路网关再与响应方建立连接，并在此过程中完成用户鉴别。这样在传输层将建立两个 TCP 连接，一个是在传输层代理和内网主机之间的 TCP 用户之间，另一个是在传输层代理与外部主机上的一个 TCP 用户之间。具体过程如图 7-6 所示。

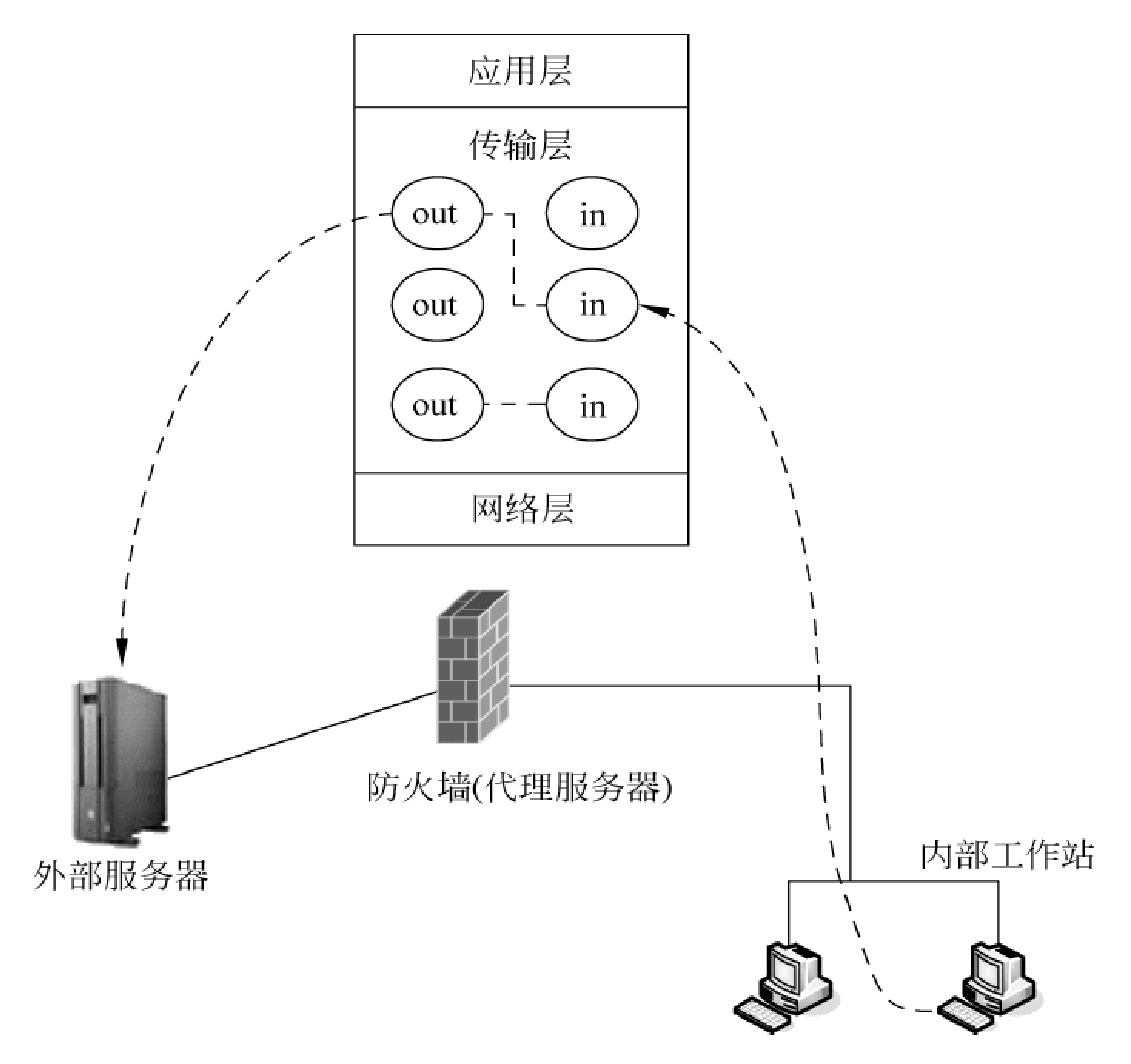


图 7-6　电路网关

电路层网关实现的一个例子是 SOCKS 软件包，SOCKS v5 在 RFC1928 定义。

SOCKS v5 不仅支持基于 TCP 连接的应用协议的代理，而且支持基于 UDP 传输的应用协议代理。它提供了一个标准的安全验证方式和对请求响应的方式。这个协议从概念上来讲是介于应用层和传输层之间的"薄层(shin-layer)"，因而不提供如传递 ICMP 信息之类由网络层网关所提供的服务。

7.2.4　网络地址转换

在 IPv4 时期，IP 地址资源短缺，对于一个机构来说，不可能为每台计算机都申请一个合法的 IP 地址，为解决这种困难，人们提出了很多方案。网络地址转换(Network Address Translation，NAT)就是一种解决方案，允许把内部私有网络地址(IP 地址)转换成合法网络 IP 地址。通过 NAT 技术，可以在局域网内使用内部 IP 地址(私有 IP 地址)，当内网的主机需要访问 Internet 时，NAT 将私有的 IP 地址转换成合法的外部地址。如图 7-7 所示，这种方式一方面可以减少对外部 IP 地址的需求，另一方面能够隐藏内部网络的结构，从而起到保护内网的作用。

随着 IPv6 的普及，IP 地址资源短缺问题将会解决，但网络地址转换功能往往在防火墙中实现，有必要对其进行了解。

1. NAT 概述

RFC 1918 规定了所谓的内部网络地址，通常这些 IP 地址用在具有 NAT 功能的防火

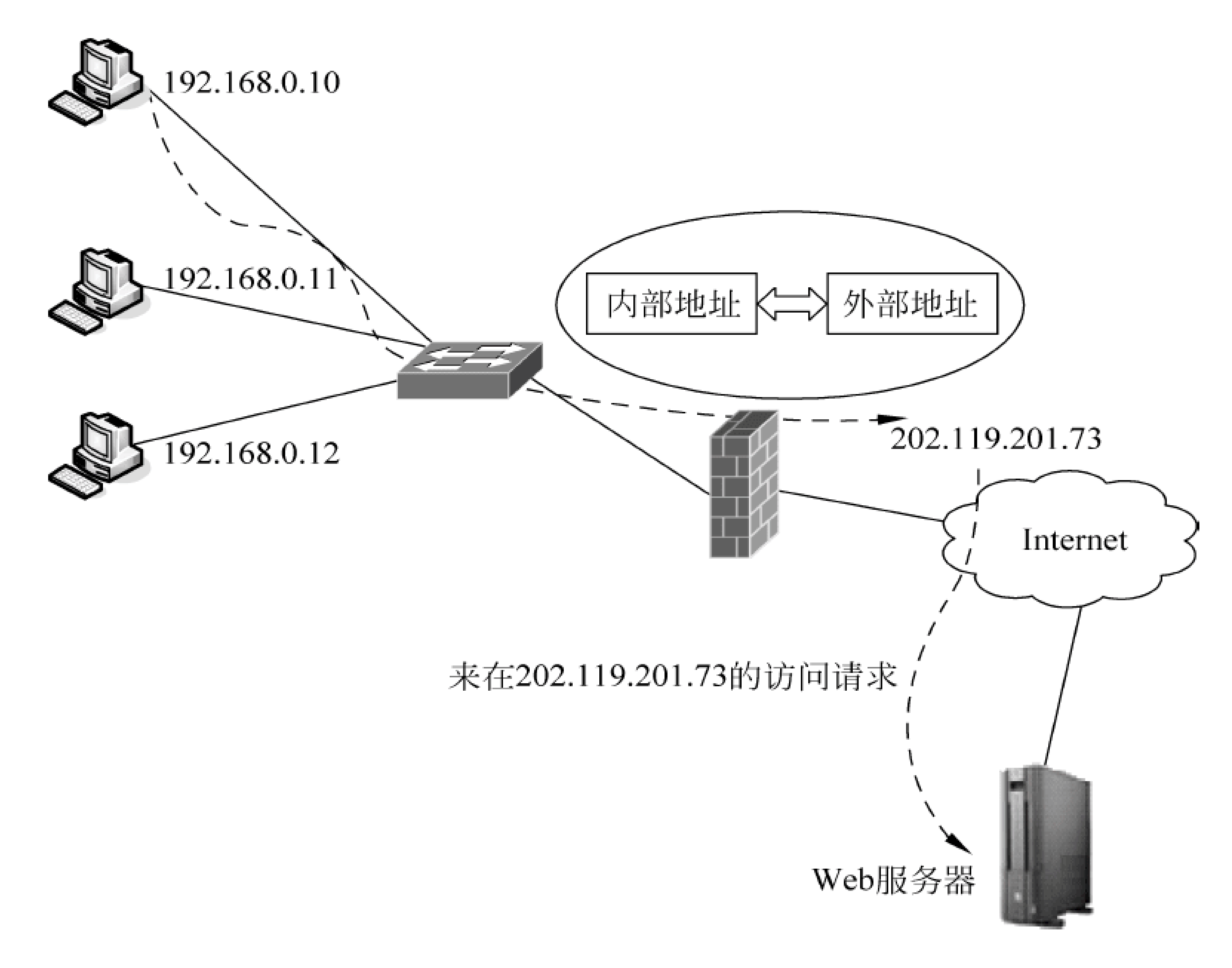


图 7-7 地址转换

墙后面的内部网络上。RFC 规定的内部地址是：10.0.0.0～10.255.255.255、172.16.0.0～172.31.255.255 和 192.168.0.0～192.168.255.255。

内部 IP 地址，只能在内部网络使用，不能在 Internet 上被寻址。

NAT 功能的实现一般是在路由器上实现的，很多防火墙也采用了 NAT 技术，设置 NAT 功能的设备至少要有一个内部端口，一个外部端口。内部端口连接的网络用户使用的是内部 IP 地址，外部端口连接的是外部的端口。

从安全的角度来说，NAT 技术不一定是为了节省 IP 地址空间。NAT 通过对外隐藏内部网络计算机的 IP 地址，为网络的安全增加了一道屏障。攻击者在扫描时，如果找不到目标机器的 IP 地址，将很难发动攻击。

NAT 设备维护一个状态表，用来把内部的 IP 地址映射到外部合法的 IP 地址上去。一般 NAT 可以分为三种类型：静态 NAT、动态 NAT 和 NAPT。NAT 的实现方式也有三种：

① M→1 多个内部地址对应一个外部地址；

② 1→1 一个内部地址对应一个外部地址；

③ M→N 多个内部网络地址对应多个外部地址。

2. 静态 NAT

静态 NAT 将内部网络的每台计算机的 IP 地址一一映射到外部、可路由的 IP 地址。内部计算机的 IP 地址仍然是隐藏的，但是这些内部网络的计算机对外部网络好像是具有公共的、可路由的 IP 地址。在静态 NAT 中，不管是公共可寻址的 IP 还是内部私有 IP 地址，都是静态的，不会动态地变化。从内部 IP 地址到公共 IP 地址的转换是静态变化的过程，如图 7-8 所示。

在图 7-8 中，每个内部 IP 地址对应一个公共 IP 地址，其中 192.168.0.10、192.168.0.

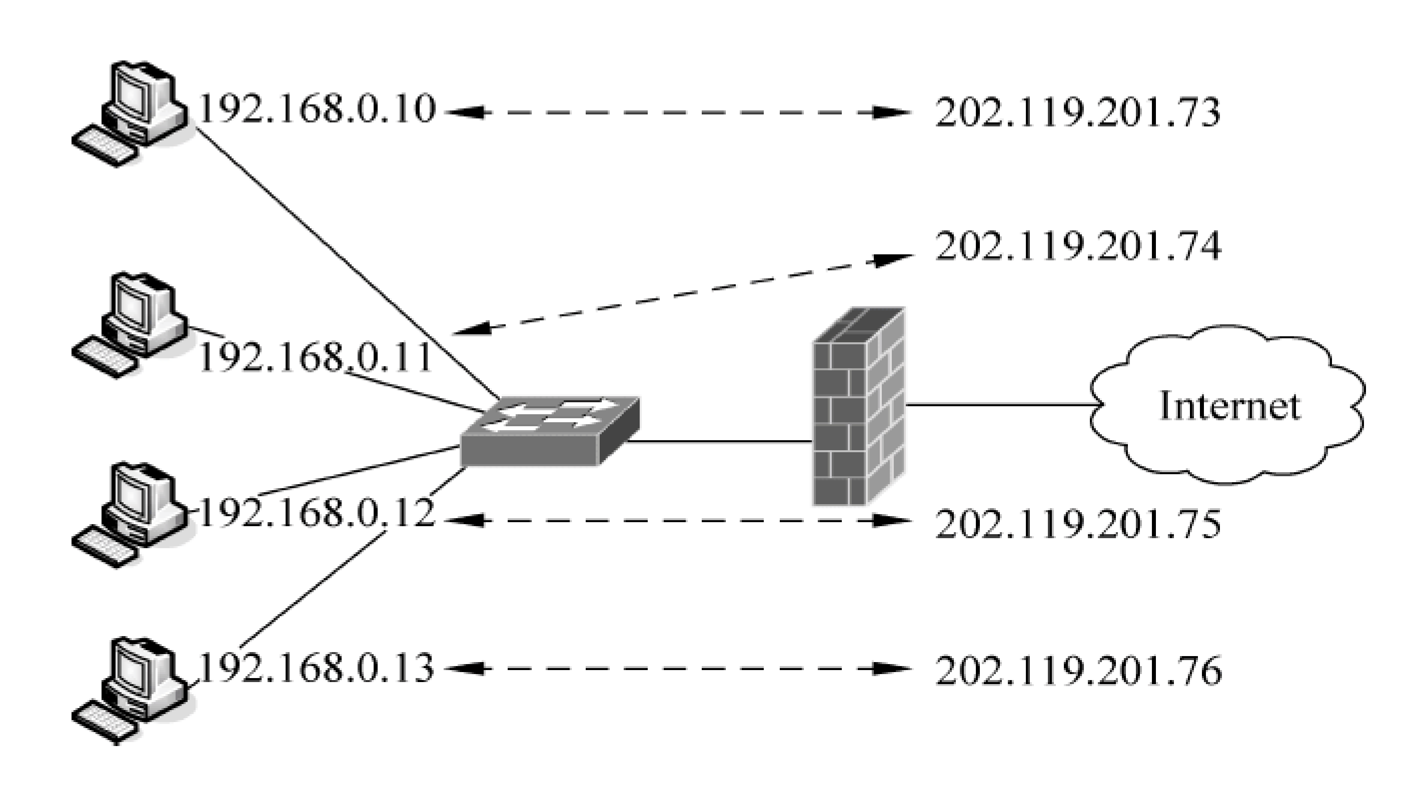


图 7-8　静态 NAT 转换

11、192.168.0.12、192.168.0.13 是内部私有地址，对应的外部公共 IP 地址为 202.119.201.73、202.119.201.74、202.119.201.75、202.119.201.76。假如，当主机 192.168.0.13 要向外部访问 Web 服务器时，就通过 NAT 转换，将内部地址转换成外部地址 202.119.201.76，与 Web 服务器建立连接；对于 Web 服务器来说好像是直接与 202.119.201.76 相连一样。

3. 动态 NAT

动态 NAT 是为每个内部的 IP 地址分配一个临时的外部 IP 地址，动态 NAT 与静态 NAT 不同的地方在于，动态 NAT 是多对多的，它将内部地址映射到一个合法地址池中。也就是说，动态 NAT 在外部网络定义了一系列的合法地址，形成地址池，地址转换时，从合法地址池中动态地选择一个未使用的地址来对内部地址进行转换。

通常，网络中的动态 NAT 设备会维护一张表，该表记录了所有可用的外部合法 IP 地址。当有内部 IP 地址请求到来时，NAT 设备会自动为其分配一个空闲的外部合法 IP 地址。例如，在图 7-9 中，当主机 192.168.0.12 想要访问 Internet 的外部服务时，NAT 设备会在地址池 202.119.201.73～202.119.201.80 中选择一个合法的 IP 地址对其进行转换。

使用动态 NAT 不仅能对外部网络屏蔽内网的部署结构，同时增加了从外部网络监控主机的难度，从而保证内网的安全。

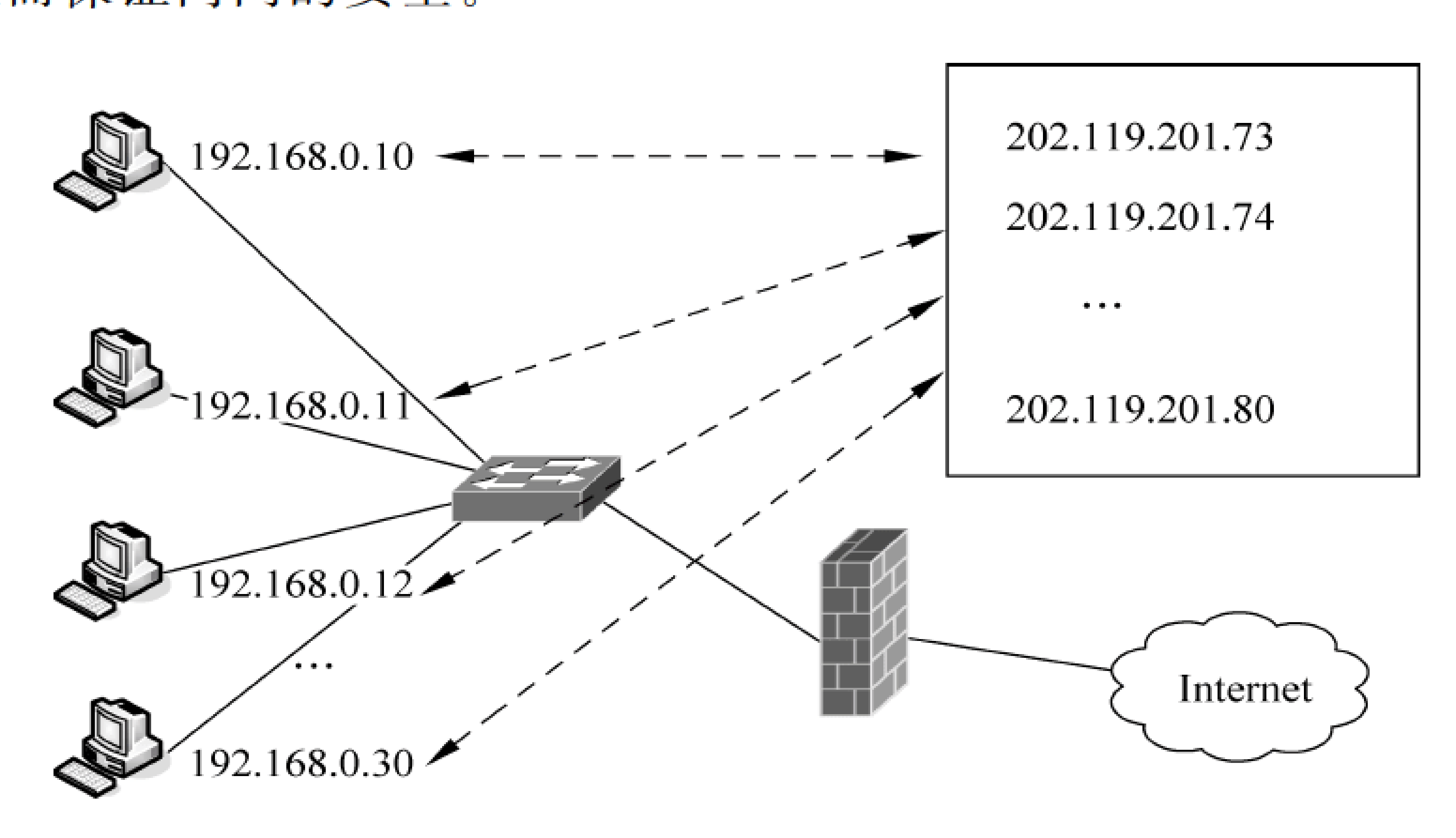


图 7-9　动态 NAT 转换

4. NAPT

在动态地址转换中,当内部网络主机访问外网的需求量增多时,合法地址列表中的 IP 地址会不够用,这时可以利用传输层 TCP/UDP 的端口号字段来协助建立 NAT 转换表项。这样多个私有地址可以通过一个合法地址进行转换。这种 NAT 技术被称为网络地址端口映射(NetWork Address Port Translation, NAPT)。

网络地址端口映射,可以使多个内部主机映射到外部网络中的一个合法的 IP 上(多对一的方式),同时在该地址加上一个由 NAT 设备选定的 TCP/UDP 端口号,再通过 TCP/UDP 的端口号来区分不同的主机连接。

例如,假设内部主机 192.168.0.11 与 192.168.0.12 都发送端口为 1200 的数据包。执行端口映射的网关可以将这两台主机发出的包的源端口改为使用两个不同端口 52001 和 52002,而源地址使用一个公用的地址 202.119.201.75。这样,目的端口为 52001 的响应包就发送给 192.168.0.11,而目的端口为 52002 的响应包发送给192.168.0.12。

5. NAT 技术的安全问题

在使用 NAT 时,Internet 上的主机表面上看起来直接与 NAT 设备通信,而非与专用网络中实际的主机通信。输入的数据包被发送到 NAT 设备的 IP 地址上,并且 NAT 设备将目的包头地址由自己的 Internet 地址变为真正的目的主机的专用网络地址。而结果是,理论上一个全球唯一 IP 地址后面可以连接几百台、几千台乃至几百万台拥有专用地址的主机。但是,这实际上存在着缺陷。例如,许多 Internet 协议和应用依赖于真正的端到端网络,在这种网络上,数据包完全不加修改地从源地址发送到目的地址。NAT 还提出了管理上的挑战。尽管 NAT 对于一个缺少足够的全球唯一 Internet 地址的组织、分支机构或者部门来说是一种不错的解决方案,但是当重组、合并或收购需要对两个或更多的专用网络进行整合时,它就变成了一个严重的问题。甚至在组织结构稳定的情况下,NAT 系统不能多层嵌套,从而造成路由噩梦。

此外,当改变网络的 IP 地址时,都要仔细考虑这样做会给网络中已有的安全机制带来什么样的影响。如防火墙根据 IP 报头中包含的 TCP 端口号、源 IP 地址、目的 IP 地址以及其他一些信息来决定是否让该数据包通过。可以按照 NAT 设备所处位置来改变防火墙过滤规则,这是因为 NAT 改变了源或目的 IP 地址。如果一台 NAT 设备,如内部路由器,被置于受防火墙保护的一侧,将不得不改变负责控制 NAT 设备后面网络流量的所有安全规则。在许多网络中,NAT 机制都是在防火墙上实现的。它的目的是使防火墙能够提供对网络访问与地址转换的双重控制功能。除非可以严格地限定哪一种网络连接可以被进行 NAT 转换,否则不要将 NAT 设备置于防火墙之外。任何一位攻击者,只要他能够使 NAT 误以为他的连接请求是被允许的,都可以以一个授权用户的身份对内部网络进行访问。如果企业正在使用 IP 安全协议(IPSec)来构造一个虚拟专用网(VPN)时,错误地放置 NAT 设备会毁了工程。原则上,NAT 设备应该被置于 VPN 受保护的一侧,因为 NAT 需要改动 IP 报头中的地址域,而在 IPSec 报头中该域是无法被改变的,这样就可以准确地获知原始报文是发自哪一台工作站的。如果 IP 地址被改变了,那么 IPSec 的安全机制也就失效了,因为既然源 IP 地址都可以被改动,那么报文内容就更不用说了。

7.2.5　匿名代理 TOR

TOR(The Onion Router)是第二代洋葱路由(onion routing)的一种实现,用户通过 TOR 可以在因特网上进行匿名交流。最初该项目由美国海军研究实验室(US Naval Research Laboratory)赞助。2004 年后期,TOR 成为电子前哨基金会(Electronic Frontier Foundation,EFF)的一个项目。2005 年后期,EFF 不再赞助 TOR 项目,但他们继续维持 TOR 的官方网站。

TOR 专门防范流量过滤、嗅探分析,使用户免受其害。TOR 在由洋葱路由器组成的表层网(overlay network)上进行通信,可以实现匿名对外连接、匿名隐藏服务。

1. 匿名外连

TOR 用户在本机运行一个洋葱代理服务器(onion proxy),这个代理周期性地与其他 TOR 交流,从而在 TOR 网络中构成虚电路(virtual circuit)。TOR 是在应用层进行加密(也就是按照 onion 的模式),而它之所以被称为 onion 是因为它的结构很像洋葱,只能看出它的外表,想要看到内核就必须把它层层剥开。即每个洋葱路由器间的传输都经过对称密钥加密,形成有层次的结构。它中间所经过的各节点,都好像洋葱的一层皮,把客户端包在里面,算是保护信息来源的一种方式,这样在洋葱路由器之间可以保持通信安全。同时对于客户端,洋葱代理服务器又作为 SOCKS 接口。一些应用程序就可以将 TOR 作为代理服务器,网络通信就可以通过 TOR 的虚拟环路来进行。

进入 TOR 网络后,加密信息在路由器间传递,最后到达"出口节点",明文数据从这个节点直接发往原来的目的地。对于目的主机而言,是从"出口节点"发来信息。

通过使用 TOR,一般的应用程序都可以实现匿名,比如 IRC、即时通信,以及浏览网页。浏览网页时,TOR 常常与 Privoxy 或 Polipo 等联合使用,Privoxy、Polipo 是开源代理服务器,可以在应用层增加保护隐私。

2. 规矩与滥用

由于 TOR 可以匿名进行 TCP 传输,这就导致了被滥用的可能。路由器存放有一个"出口政策"(exit policy),它含有各种地址、端口的组合规定,通过这个来区别哪些传输可以通过这个节点而离开 TOR 网络,这样就可能防范许多滥用的可能。潜在的滥用包括以下几种。

- P2P:抛开合法性不谈,如果通过 TOR 网络来进行巨额数据的传输,那就不合适了,毕竟洋葱路由器是由志愿者花费自己的带宽建立起来的。
- E-mail:匿名的 SMTP 很容易导致垃圾邮件的产生,一般 TOR 节点的"出口政策",都拒绝对外连接到端口 25(smtp 的端口)。
- 蓄意破坏:由于不会被识破,用户有时会利用 TOR 来对协作网站进行破坏,这导致许多的网站决定部分限制对 TOR 的通信。

3. 隐蔽的匿名服务和网站

TOR 不仅可以提供客户端的匿名访问,TOR 还可以提供服务器的匿名。通过使用 TOR 网络,用户可以维护位置不可知的服务器。当然如果要访问这个隐蔽的服务,客户端也得安装 TOR。

通过 TOR 指定的顶级域名(Top Level Domain,TLD).onion,可以访问其隐藏的服务。TOR 网络可以识别自己的 TLD,并自动路由到隐藏的服务。然后,隐藏的服务将请求交由标准的服务器软件进行处理,这个服务器软件应该预先进行配置,从而只侦听非公开的接口。如果这个服务还可以通过公共的因特网来访问,那也会受到相关连的攻击,这样就没有真正的隐藏起来。TOR 隐藏服务有个另外的好处,由于不需要公开的 IP 地址,服务就可以躲在防火墙和 NAT 背后。

4. 反制防火墙

TOR 被许多人用来突破防火墙,这主要是因为防火墙屏蔽了大量的网站。TOR 会自动检测节点是否可达目标地址。如果返回错误,它会自动更换节点。

7.3 防火墙的体系结构

在一个网络系统中,防火墙是一个进行安全防护的系统,由于网络结构和网络设备的多样性,防火墙在网络中的配置体系结构可能多种多样,在现有的防火墙配置结构中,主要有屏蔽路由器、双宿主主机防火墙、屏蔽主机防火墙与屏蔽子网防火墙四种模式。在介绍四种模式之前,首先了解一下堡垒主机的概念。

堡垒主机是一种被强化的可以防御进攻的计算机,它对外部网络是暴露的,它为网络间的通信提供了一个阻塞点。通常堡垒主机作为应用网关和电路网关的平台,是一个组织机构网络安全的中心主机。因此,经常是攻击者的主要攻击对象,对它必须进行完善的防御。网络管理员必须严密监视堡垒主机,堡垒主机软件和系统的安全情况应定期进行检查,对访问记录应进行查看,以防潜在的安全威胁和漏洞。

7.3.1 屏蔽路由器结构

屏蔽路由器结构(包过滤路由器防火墙)是一种比较简单的防火墙结构,通过在内部网络和外部网络的关键路径上设置一台带有包过滤功能的路由器实现对网络系统的保护。屏蔽路由器作为内外连接的唯一通道,要求所有的网络数据包都必须在此通过检查。防火墙允许通信的内、外网通信的主机之间进行直接的报文交换,具有一定的安全风险,如果防火墙被渗透或攻破,整个内部网络将完全暴露在攻击者面前。

屏蔽路由器结构的防火墙容易实现、配置简单、对用户透明,适用于小型、不太复杂的网络系统。屏蔽路由器的缺点是:日志记录能力差、规则表达式比较复杂,可能导致出现逻辑差错;不能阻止某些类型的攻击,特别是采用地址欺骗攻击技术的攻击,而且屏蔽路由器一旦被攻克后很难发现攻击者,而且不能识别不同的用户。

7.3.2 双宿主主机防火墙

双宿主主机模式防火墙(dual-homed firewall),又称双宿网关防火墙。在这种防火墙配置形式中,通常使用一台装有两块或多块网卡的堡垒主机作为网关,堡垒主机上装有代理服务器软件(防火墙软件),两块网卡分别连接内网和外网,如图 7-10 所示。

在双宿主主机防火墙模式下,堡垒主机关闭了 IP 数据包转发功能,采用代理的服务方

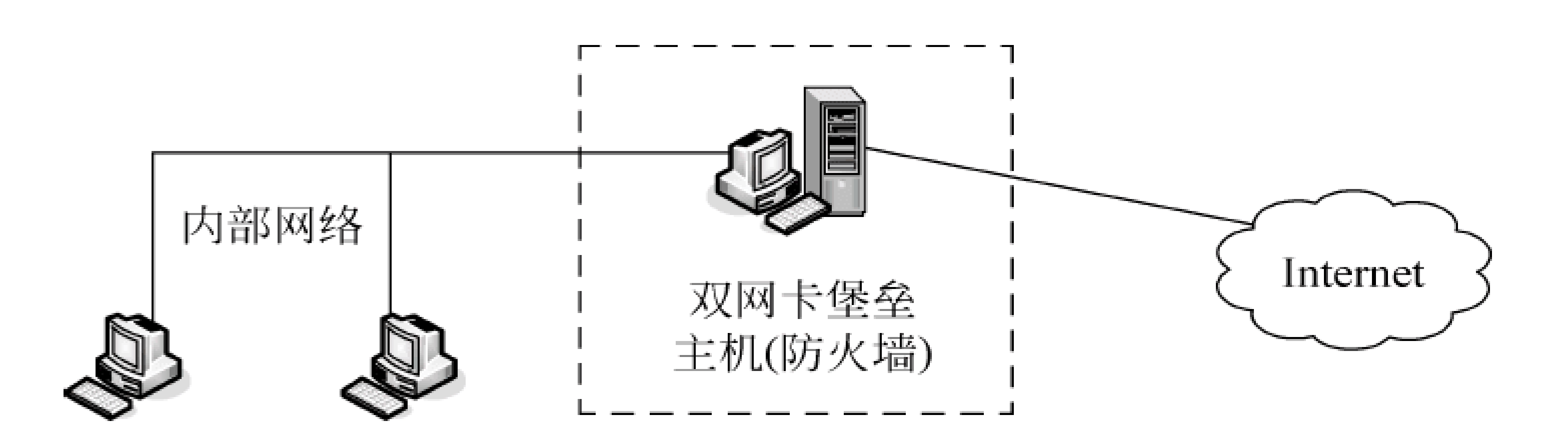


图 7-10　双宿主主机模式防火墙

式，堡垒主机相当于一个应用网关或电路网关。内部主机从外部不可见，由防火墙按照安全策略中继允许的网络服务。

双宿主主机防火墙，结构简单明了，容易实现，成本较低，将内部子网和外部 Internet 采用物理设备和代理方式的双重隔离，内部网络的信息不会泄露到外界，内部主机对外部网络是隐蔽的；提供了强大的审计能力和鉴别服务以及应用代理服务。

这种模式的防火墙的弱点是，对所有的服务协议和应用都需要由代理服务器来完成，网络访问速度降低，不适合内部子网大量访问外部网络的场合；若提供防火墙功能的堡垒主机被黑客攻破，打开其 IP 数据包转发功能，则任何网上用户均可随意访问其内部网络。

7.3.3　屏蔽主机模式防火墙

屏蔽主机模式防火墙是一种包过滤路由器加应用层网关的双重安全保障体系，而且具有灵活配置的防火墙方式，如图 7-11 所示。屏蔽主机模式的防火墙由包过滤路由器和堡垒主机(应用网关)组成。

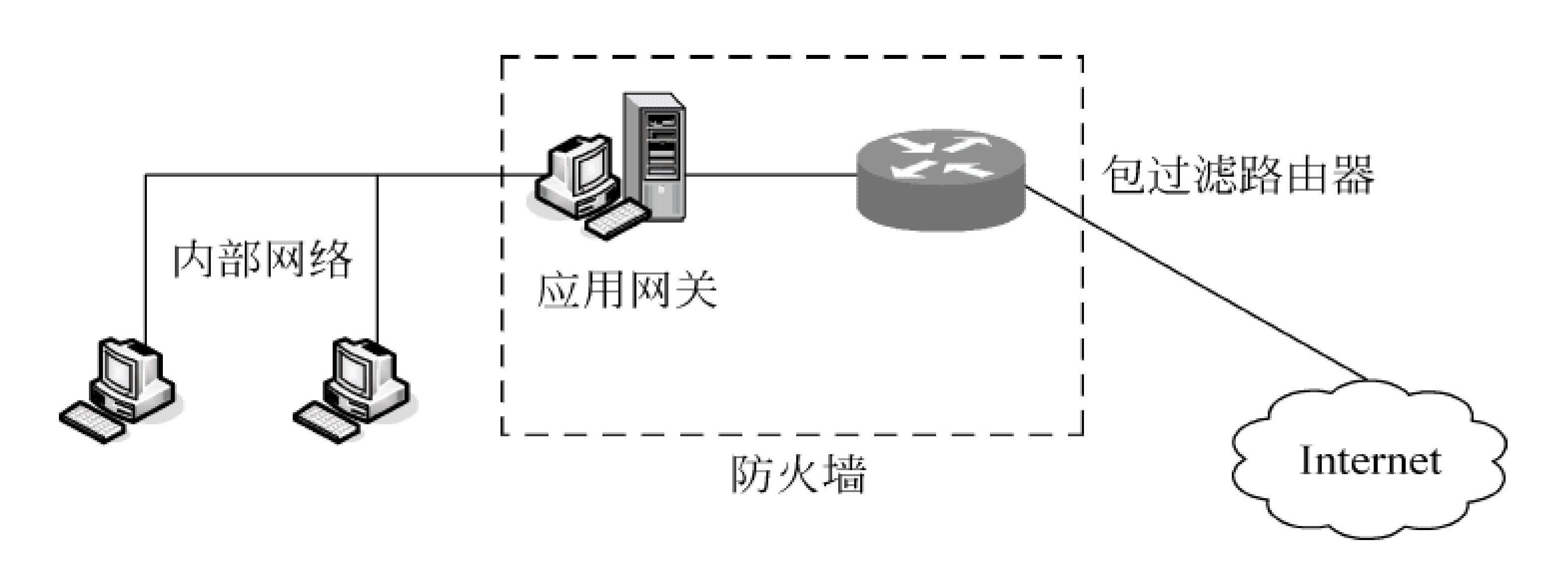


图 7-11　屏蔽主机防火墙

该模式的防火墙中，堡垒主机仅与内网相连，而包过滤路由器连通内部网和外部网。任何来自外部网络的所有通信先到达包过滤路由器，包过滤路由器根据其配置的规则对通信进行过滤，与应用网关的代理服务无关的通信将被拒绝。应用网关的代理服务器将包过滤路由器允许的外部网络通信传送给被保护的内部网络。另一方面，内部向外的访问可能通过应用网关代理，再经包过滤路由器到达目标。所有的网络流量都必须通过堡垒主机，堡垒主机需要拥有高等级的安全。因此在屏蔽的路由器中数据包过滤配置应满足：

(1) 对于来自外部网络的网络流量，只有发往堡垒主机的 IP 数据包才被允许通过；

(2) 对于来自内部网络的网络流量，只有来自堡垒主机的 IP 数据包才被允许通过。

屏蔽主机模式防火墙可以实现数据包级过滤和应用级的过滤，比单独使用包过滤或应用网关防火墙更加安全。配置比较灵活，在这种模式下，内部网络用户访问外部网络较为方便和灵活，在包过滤路由器和堡垒主机允许的情况下，用户可以直接访问外部网络。

屏蔽主机模式防火墙的缺点：如果攻击者一旦攻破路由器后面的堡垒主机，就会导致内部网络处于危险状态；两个防火墙部件的配置在逻辑上要求严密准确，配置比较复杂，容易出现错误。

7.3.4 屏蔽子网模式防火墙

在双宿主主机和屏蔽主机模式的防火墙体系结构中，堡垒主机都是一个安全缺陷，一旦堡垒主机被攻破，则整个的网络都处于威胁之中，解决的办法就是增加一个包过滤路由器，即使用屏蔽子网模式防火墙。

屏蔽子网就是在内部网络和外部网络之间建立一个被隔离的子网，用两台包过滤路由器将这一子网分别与内部网络和外部网络分开。如图 7-12 所示的一种实现，两个包过滤路由器放在子网的两端，在子网内构成一个“非军事区(DMZ)”，内部网络和外部网络均可访问被隔离的子网，但禁止它们穿过被屏蔽子网通信。有的屏蔽子网中还设有一台堡垒主机对外提供公共服务，如 WWW 服务、FTP 服务以及 DNS 服务等。为了侵入用这种类型的体系结构构筑的内部网络，侵袭者必须要通过两个包过滤路由器。即使侵袭者设法侵入堡垒主机，它将仍然必须通过内部路由器。

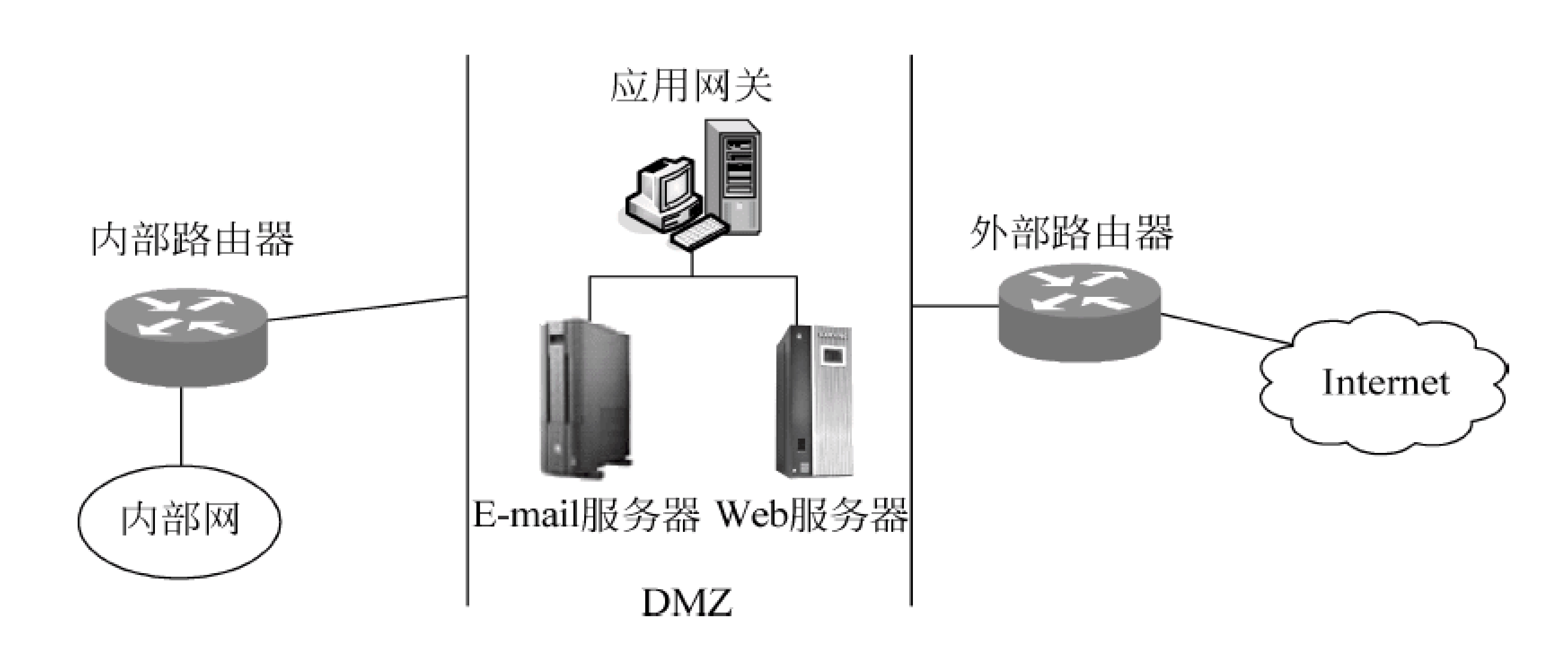


图 7-12 屏蔽子网模式防火墙

屏蔽子网模式防火墙，安全强度高，攻击者要进入被保护的内部网络几乎不可能；但同时投资较高，对安全管理员的要求较高，两个包过滤路由器在配置上很复杂。

7.4 防火墙的局限性与发展趋势

7.4.1 防火墙的局限性

尽管防火墙可以对网络的安全起到保护作用，但不能保证网络的绝对安全，在功能和性能上都有一定的局限性，列举如下。

1. 防火墙一般不能防范内部主动发起的攻击

防火墙是一种实施隔离措施的系统，对外部具有严格的访问限制措施，但对内部发起的攻击却无能为力。

2. 防火墙不能抵抗最新的未知攻击

防火墙是一个被动的安全策略执行设备，对于新的未知攻击或策略配置错误，也无能为力。如杀毒软件和病毒一样，总是先出现病毒，经过分析人员分析出特征码加入到病毒库库内才能查杀。防火墙的各种策略，也是在该攻击方式经过专家分析后给出其特征进而设置的。

3. 防火墙对于利用网络协议缺陷进行的攻击无法防范

防火墙作为内网与外网的通信中介，必然需要允许网络协议的执行，如果攻击者利用存在缺陷的协议进行攻击，防火墙不能避免这种攻击。例如，利用 DoS 或 DDoS 进行攻击。

4. 防火墙对服务器合法开放的端口的攻击大多无法阻止

防火墙不能防止黑客通过防火墙准许的访问端口对该服务器的漏洞进行攻击，攻击者可利用服务器提供的服务进行缺陷攻击。

5. 防火墙本身会出现安全漏洞和受到攻击

防火墙自身也是一个系统，也有着其硬件系统和软件，因此依然有着漏洞，所以其本身也可能受到攻击和出现软、硬件方面的故障。没有厂商绝对保证防火墙不会存在安全漏洞。

6. 防火墙不能防止数据驱动式的攻击

当有些表面看起来无害的数据通过邮件或复制到内部网的主机上并被执行时，就会发生数据驱动式的攻击。例如，一种数据驱动式的攻击造成主机修改与系统安全有关的配置文件，从而使入侵者下一次更容易攻击该系统。

7. 防火墙不能防止感染了病毒的软件或文件的传输

防火墙本身不具备查杀病毒的功能，即使集成了第三方的杀毒软件，通常也不能及时阻止病毒的传播和入侵。

8. 防火墙可以阻断攻击，但不能消灭攻击源

互联网上病毒、木马、恶意试探等造成的攻击行为络绎不绝。设置得当的防火墙能够阻挡它们，但是无法清除攻击源。即使防火墙进行了良好的设置，使得攻击无法穿透防火墙，但各种攻击仍然会源源不断地向防火墙发出尝试。

9. 防火墙不能防范人为因素的攻击

防火墙不能防止由内部人员误操作或恶意破坏造成的威胁，防火墙不能防止用户由于口令泄露而遭受到的攻击。

此外，防火墙在性能上不具备实时监控入侵的能力；其功能和速度成反比，防火墙的功能越多，对 CPU 和内存的消耗就越大，速度越慢。因此，在构建网络防御系统时，防火墙是一个防护措施，但仅仅依靠防火墙是不够的。

7.4.2　防火墙的发展趋势

针对现有防火墙技术存在的不足，未来防火墙的发展趋势是朝高速度、智能化、多功能化、人性化的方向发展。

1. 速度上的发展

目前防火墙一个很大的局限性是速度不够，应用网络处理器是实现高速防火墙的主要

方法。实现高速防火墙,算法也是一个关键,因为网络处理器集成了很多硬件协处理单元,因此比较容易实现高速。对于采用纯 CPU 的防火墙,就必须有算法支撑。

2. 智能化的发展

防火墙将向着具备人工智能的智能化方向发展。未来智能防火墙应具备自主学习及制定识别与防御方法的能力,能自动识别并防御各种黑客攻击手法及其相应变种攻击手法;能在网络出口发生异常时自动调整与外网的连接端口;能根据信息流量自动分配、调整网络信息流量及协同多台物理设备工作;具有自动检测及自动修复功能。

3. 功能上的发展

多功能也是防火墙的发展方向之一,目前的组网环境比较复杂,一般用户希望防火墙可以支持更多的功能来满足多样化的组网需求,降低用户对其他专用设备的需求,节省投资成本。

4. 人性化的发展

未来的防火墙要具有易于安装和配置、管理的图形化界面,减少因配置管理而导致的设定、实现的错误。

习　题　7

1. 什么叫防火墙?说明防火墙的作用和设置防火墙的目的。
2. 防火墙的功能有哪些?
3. 防火墙主要分为哪几类?
4. 说明包过滤防火墙存在的缺陷或问题。
5. 简述应用层代理的工作原理。
6. 如何访问受限站点?
7. 说明 Internet 防火墙的应用体系结构及其主要特点。
8. 你是如何配置个人防火墙的?
9. 防火墙的局限性主要有哪些表现?

第8章　系统入侵检测与防御

8.1　入侵检测系统

8.1.1　入侵检测系统的概念

随着网络的发展和应用的日益广泛,人们发现仅仅依靠传统的操作系统加固防火墙等静态安全防御技术已经无法满足现有网络安全的需要。入侵检测系统(Intrusion Detection System,IDS)作为一种积极主动防御的安全保护技术,对各种被动保护技术起到了极其有益的补充。

入侵检测的概念最早由 Anderson 在 1980 年提出,是指对入侵行为的发觉,并对此做出反应的过程。入侵检测是防火墙的合理补充,帮助系统对付网络攻击,扩展系统管理员的安全管理能力(包括安全审计、监视、进攻识别和响应),提高信息安全基础结构的完整性。它从计算机网络系统中的若干关键点收集信息,并分析这些信息,查看网络中是否有违反安全策略的行为和遭到袭击的迹象。入侵检测被认为是防火墙之后的第二道安全闸门,在不影响网络性能的情况下对网络进行监测,从而提供对内部攻击、外部攻击和误操作的实时保护。

入侵检测系统主要有以下的功能:

(1) 监视并分析用户和系统的活动;

(2) 检查系统配置和漏洞;

(3) 识别已知的攻击行为并报警;

(4) 异常行为模式的统计分析;

(5) 评估系统关键资源和数据文件的完整性;

(6) 操作系统的审计跟踪管理,并识别用户违反安全策略的行为。

入侵检测系统的单独使用不能起到保护系统和网络的作用,也不能单独地防止任何攻击。但它是整个系统的主要组成部分,扮演一个侦察和预警的角色,协助管理员发现和处理已知攻击和异常行为。

8.1.2　基于主机的入侵检测系统

基于主机的入侵检测系统(Host-based Intrusion Detection System,HIDS)为早期的入侵检测系统,其检测的目标主要是主机系统和系统本地用户。检测原理是根据主机的审计数据和系统的日志发现可疑事件,检测系统可以运行在被检测的主机或单独的主机上,基本过程如图 8-1 所示。

基于主机的 IDS 使用验证记录,并发展了精密的可迅速做出响应的检测技术。通常,

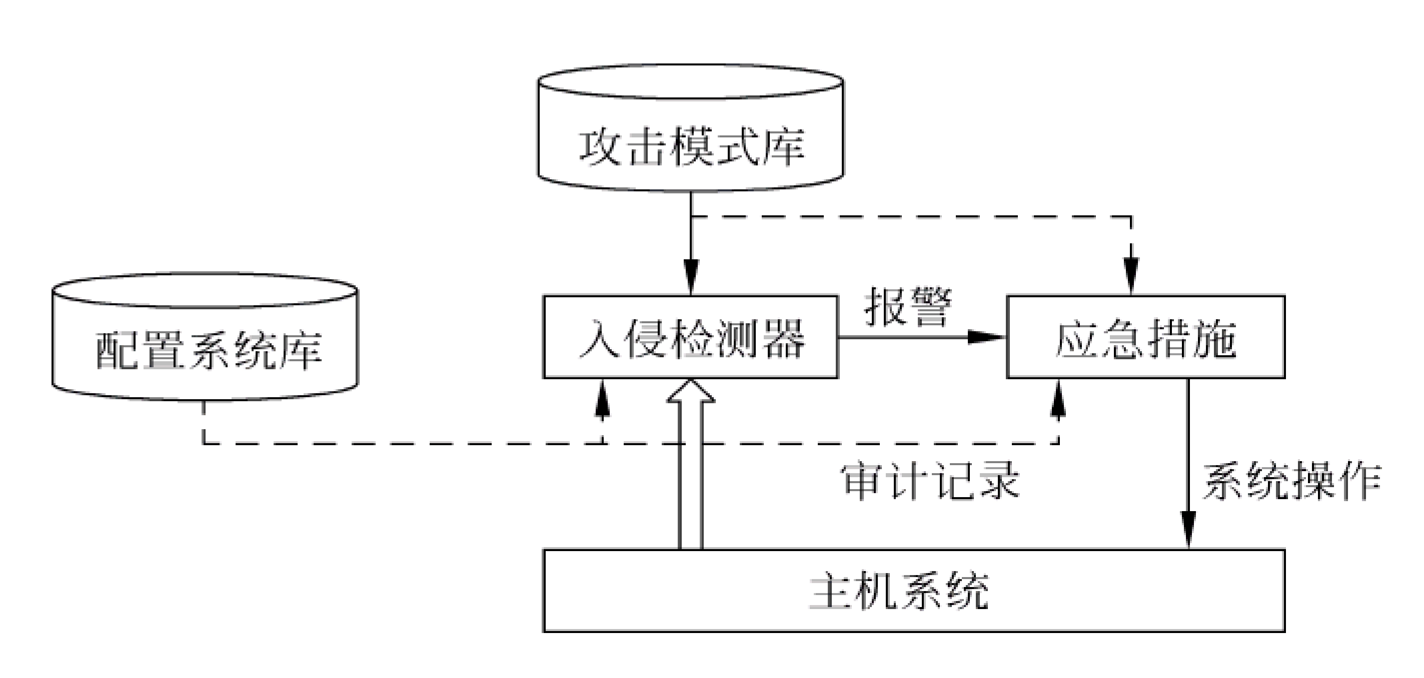


图 8-1 基于主机系统的结构

基于主机的 IDS 可监探系统、事件和 Windows 下的安全记录以及 UNIX 环境下的系统记录。当有文件发生变化时，IDS 将新的记录条目与攻击标记相比较，看它们是否匹配。如果匹配，系统就会向管理员报警并向别的目标报告，以采取措施。

基于主机的 IDS 在发展过程中融入了其他技术。对关键系统文件和可执行文件的入侵检测的一个常用方法，是通过定期检查校验和来进行的，以便发现变化。许多系统都是监听端口的活动，并在特定端口被访问时向管理员报警。这类检测方法将基于网络的入侵检测的基本方法融入到基于主机的检测环境中。

基于主机的入侵检测系统依赖于审计数据或系统日志准确性和完整性以及安全事件的定义。若入侵者设法逃避审计或进行合作入侵，则基于主机检测系统就暴露出其弱点，特别是在现在的网络环境下，单独地依靠主机审计信息进行入侵检测难以适应网络安全的需求。

基于主机的入侵检测系统有以下的弱点。

(1) 主机入侵检测系统安装在需要保护的设备上，如当一个数据库服务器要保护时，就要在服务器上安装入侵检测系统，这会降低应用系统的效率。此外，它也会带来一些额外的安全问题，安装了主机入侵检测系统后，将安全管理员本来无权访问的服务器变成他可以访问的了。

(2) 主机的审计信息弱点，如易受攻击，入侵者可通过使用某些系统特权或调用比审计本身更低级的操作来逃避审计。

(3) 主机入侵检测系统依赖于服务器固有的日志与监视能力，且只能对服务器的特定的用户、应用程序执行动作，对日志进行检测，所能检测到的攻击类型受到限制。

(4) 主机入侵检测系统除了监测自身的主机以外，根本不监测网络上的情况，不能通过分析主机审计记录来检测网络攻击(如域名欺骗、端口扫描等)。

(5) 全面部署主机入侵检测系统代价较大，企业中很难将所有主机用主机入侵检测系统保护，只能选择保护部分主机。那些未安装主机入侵检测系统的机器将成为保护的盲点，入侵者可利用这些机器达到攻击目标。

8.1.3 基于网络的入侵检测系统

随着计算机网络技术的发展，单独地依靠主机审计信息进行入侵检测难以适应网络安全的需求。从而人们提出了基于网络入侵检测系统体系结构，这种检测系统根据网络流量、单台或多台主机的审计数据检测入侵。基于网络的入侵检测系统(Network Intrusion Detection System，NIDS)放置在比较重要的网段内，不停地监视网段中的各种数据包。对

每一个数据包进行特征分析,如果数据包与系统内置的某些规则吻合,入侵检测系统就会发出警报甚至直接切断网络连接。目前,大部分入侵检测系统是基于网络的。

图 8-2 中的探测器由过滤器、网络接口引擎器以及过滤规则决策器构成,探测器的功能是按一定的规则从网络上获取与安全事件相关的数据包,然后传递给分析引擎器进行安全分析判断。分析引擎器将从探测器上接收到的包结合网络安全数据库进行分析,把分析的结果传递给配置构造器。配置构造器按分析引擎器的结果构造出探测器所需要的配置规则。

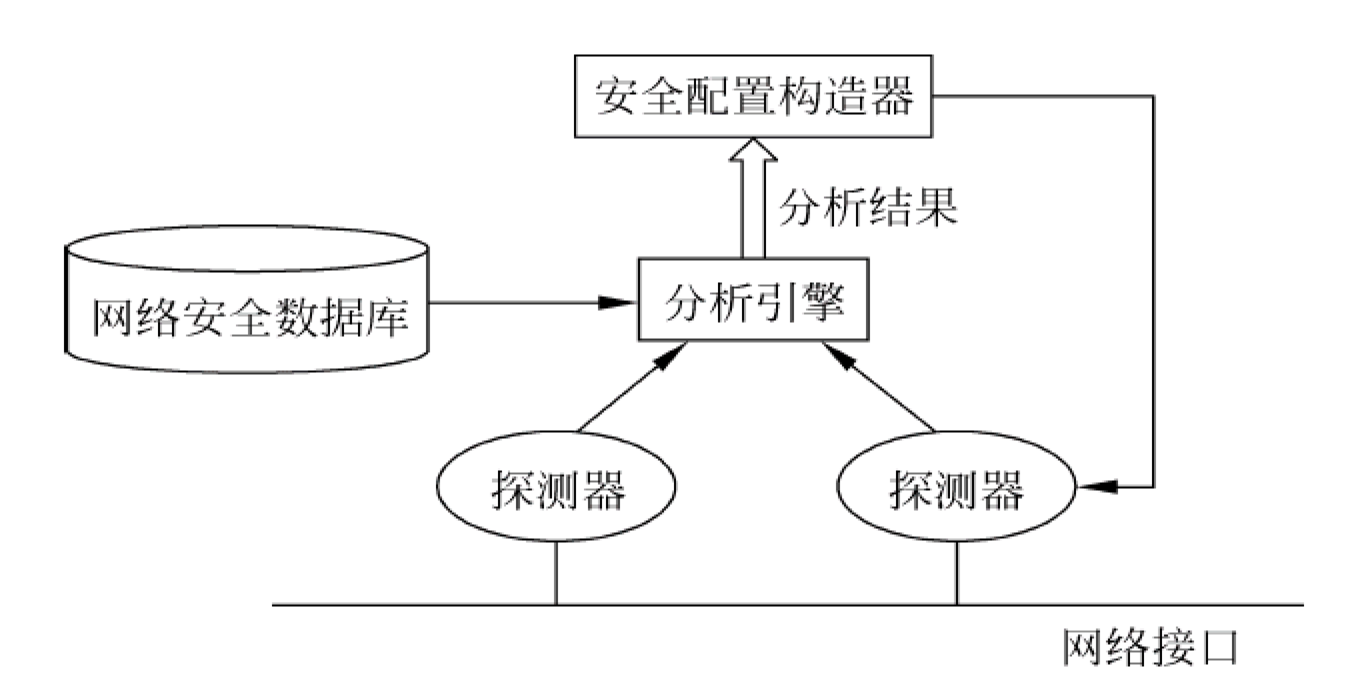


图 8-2　基于网络系统的结构

基于网络的 IDS 已经广泛成为安全策略的实施中的重要组件,它有许多仅靠基于主机的入侵检测法无法提供的优点。

(1) 平台无关性。基于网络的 IDS 作为安全监测资源,与主机的操作系统无关,监视通信量而不影响服务器平台的变化和更新。与之相比,基于主机的系统必须在特定的、没有遭到破坏的操作系统中才能正常工作,生成有用的结果。

(2) 成本低、配置简单。基于网络的 IDS 可在几个关键访问点上进行策略配置,以观察发往多个系统的网络通信,所以它不要求在许多主机上装载并管理软件。由于需监测的点较少,因此对于一个公司的环境来说,花费成本很低。

(3) 检测的攻击标记标识较多。基于网络的 IDS 探测器检查所有包的头部从而发现恶意的和可疑的行动迹象。基于主机的 IDS 无法查看包的头部,所以它无法检测到这一类型的攻击。例如,许多来自于 IP 地址的拒绝服务型和碎片型攻击经过网络时,都可以在基于网络的 IDS 中通过实时监测包流而被发现。

(4) 实时性的检测和响应。基于网络的 IDS 可以在恶意及可疑的攻击发生时,及时将其检测出来,并做出更快的通知和响应,从而将入侵活动对系统的破坏减到最低。

网络入侵检测系统的缺点有:

(1) 不适用于交换的网络环境。在这种网络环境下,IDS 的检测范围存在局限性,用户需要添加额外的设备扩大检测范围。

(2) 网络入侵检测系统不适用于加密的网络环境。NIDS 一般不涉及对加密数据包的解密工作,也无法监视加密数据包的内容,特别是对加密病毒的攻击和入侵无能为力。目前通过加密通道的攻击尚不多,但随着 IPv6 的普及,这个问题会越来越突出。

(3) 网络入侵检测系统为了性能目标通常采用特征检测的方法,它可以检测出普通的一些攻击,而很难实现一些复杂的需要大量计算与分析时间的攻击检测。

(4) 网络流量较大时，处理能力有限。网络入侵检测系统可能会将大量的数据传回分析系统中。在一些系统中监听特定的数据包会产生大量的分析数据流量。一些系统在实现时采用一定方法来减少回传的数据量，对入侵判断的决策由传感器实现，而中央控制台成为状态显示与通信中心，不再作为入侵行为分析器。

对于 HIDS 来说，其部署方式比较简单，只需要将其安装在需要保护的主机上即可。但对于 NIDS 来说，它的部署方式就比较复杂。这不但与网络本身的拓扑结构有关，而且还与用户希望达到的监控目标有关。例如，网络管理员希望监控所有进入内部网络的数据包，那么在只有一个外部接口的情况下，只需要将 NIDS 放置在防火墙或路由器之后即可。但如果网络中存在多个外部接口，则需要在每个接口处放置一个 NIDS。图 8-3 为一种 NIDS 部署示意图。

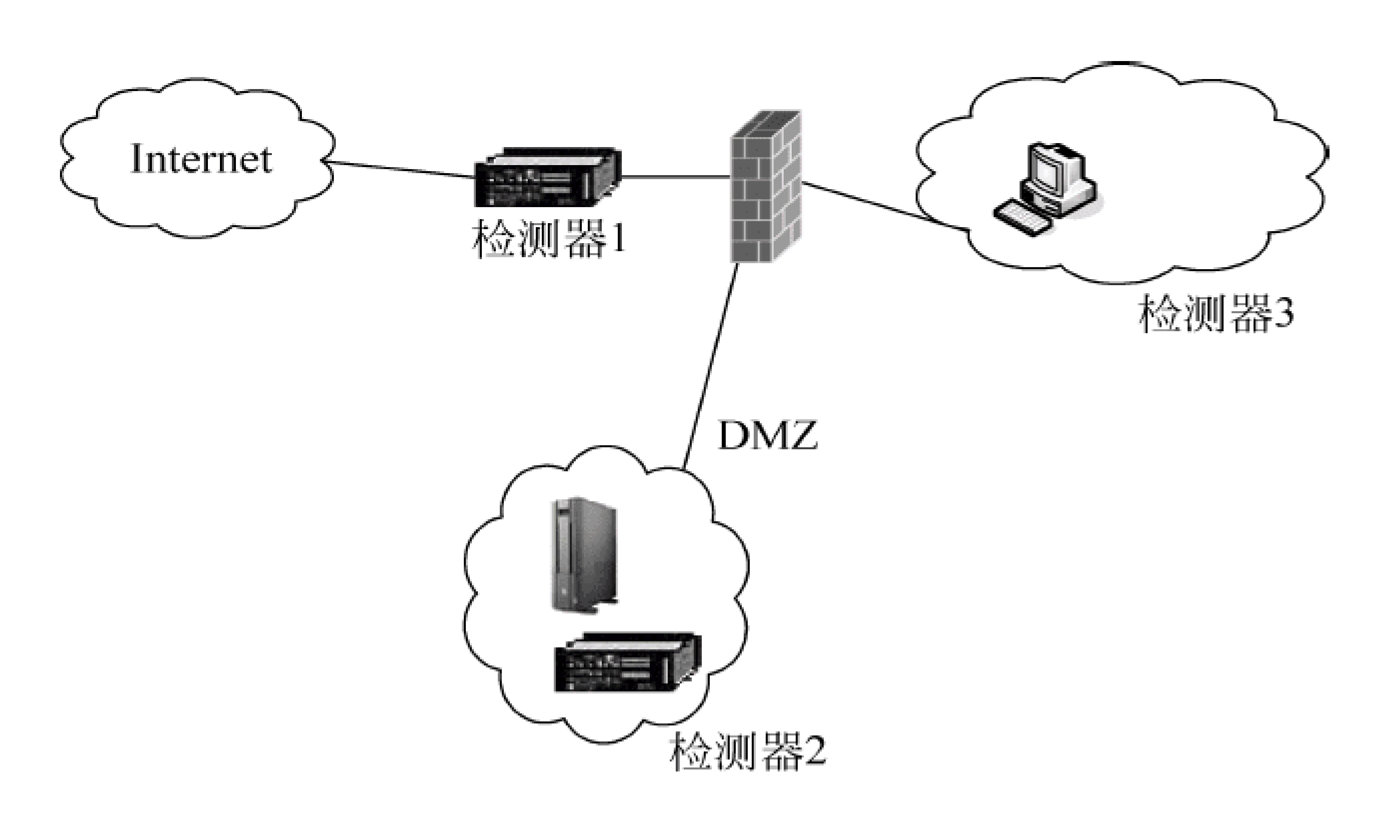


图 8-3 NIDS 部署示意图

(1) 检测器 1 位于防火墙之外，它能够检查所有到达和离开内部网络的数据，包括被防火墙阻拦的数据、进入和离开内部网络的数据。这样的 IDS 可以用来保护防火墙、抵御拒绝服务攻击，以及用来在网络中产生噪声数据的工具。

(2) 检测器 2 位于 DMZ 区，它负责检查流入此区域的数据，用来观察哪些用户试图获得 DMZ 区的访问。

(3) 检测器 3 位于防火墙内，故只能监视防火墙允许通过的数据和内部网络数据，它所报告的入侵需要网络管理员立刻注意和响应。这种针对防火墙内部接口部署网络 IDS 是一种最佳实践。

除了放置位置外，另外一个需要注意的问题是如何处理交换式的网络环境。在交换式的以太网中，由于交换机是非共享介质，它会根据数据包的目的 MAC 地址将其发送到相关的端口，而不是像集线器那样向所有端口转发数据包。这个特点使得入侵检测系统无法捕捉到达某个交换机的所有数据包。通常有两种解决方案。

一种方法是使用带调试端口的交换机。在这种交换机中，任何从其他端口进出的数据包都可以从调试端口处得到。现在已经有很多交换机都具有此功能。图 8-4 是一种典型的应用场景，为了监控防火墙和交换机之间的网络数据，将 NIDS 所在的主机的网卡接到交换机的调试端口上。

另一种方法是使用 HUB，将 HUB 放置在需要被监控的连接之间，这通常发生在路由器与交换机、交换机与交换机以及服务器与交换机之间。对于前一个例子，为了截获防火墙

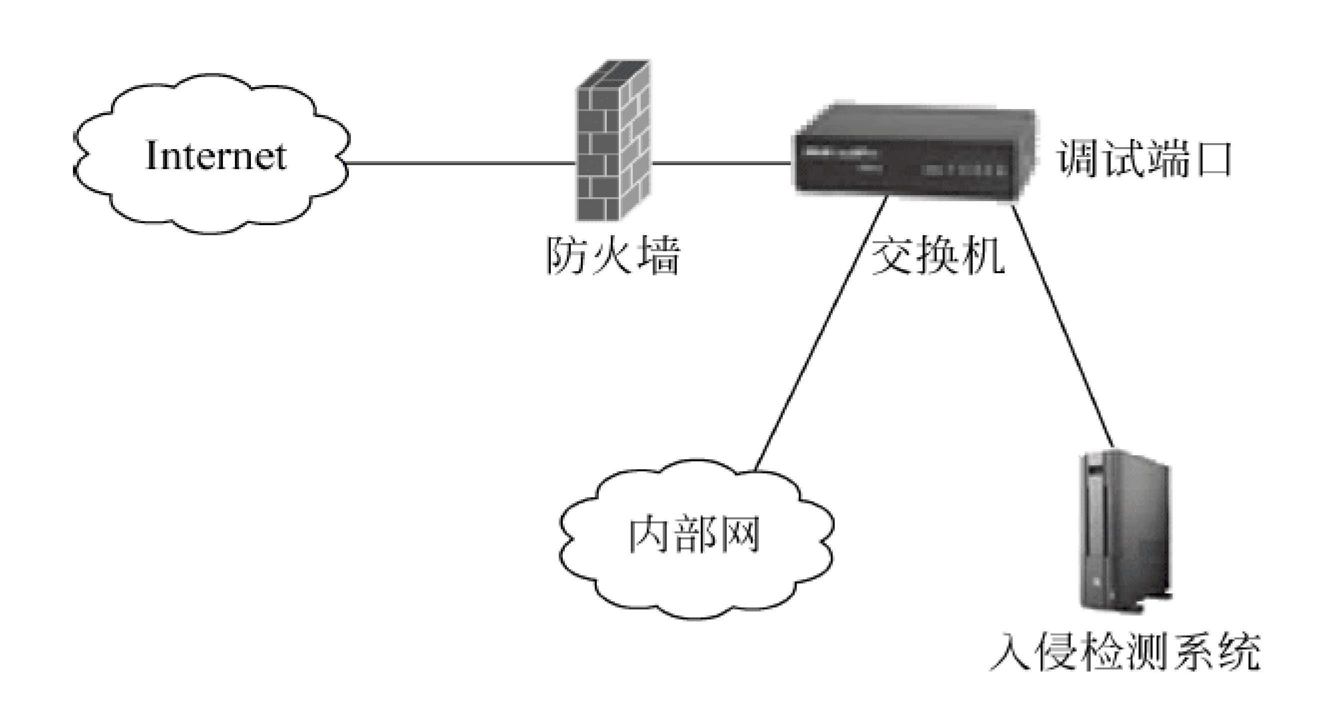


图 8-4　利用交换机调试端口捕获网络中的数据包

和交换机之间的网络通信，可以用 HUB 将防火墙和交换机连接起来，并将 NIDS 所在的主机的网卡连接到 HUB 的其中一个端口上。

8.1.4　混合型入侵检测系统

基于主机的 IDS(HIDS)和基于网络的 IDS(NIDS)各有优缺点，且具有互补性。NIDS 能够独立于主机，不影响主机性能，并且监控范围广，能够客观地反映网络活动，特别是能够监视到系统审计的盲区；而 HIDS 能够更加精确地监视系统中的各种活动，不会因为网络流量的增加而放弃对网络行为的监视，并且可以用于加密环境。因此，为了克服两者的不足造成防御体系的不全面，20 世纪 90 年代以后，许多大型的分布式入侵检测系统都是混合型的入侵检测系统。混合型入侵检测系统集 HIDS 和 NIDS 的优点于一身，即可以发现网络中的攻击信息，也可以从系统日志中发现异常情况，从而大大提高了入侵检测系统的功能。

8.2　入侵响应

8.2.1　被动响应入侵检测系统

被动响应系统只会发出告警通知，将发生的异常情况报告给管理员，本身并不试图降低所造成的破坏，更不会主动地对攻击者采取反击行动，而由管理员决定是否采取下一步行动。告警和通知、SNMP 陷阱和插件是两种常用的被动响应技术。告警和通知是最常见的被动响应方式。入侵检测系统一般提供多种形式的告警生成方式以供选择，允许用户设置告警以适合本组织的系统操作程序规范。一种是告警显示屏，这是 IDS 最常用的告警和通知方式，这种告警消息显示在 IDS 控制台上，或在由用户配置的其他系统上。另一种是远程通知，比如短消息、E-mail 等。这种方式比较方便、灵活，对于管理员及时处理突发事件比较有用。另一种被动响应方式是 SNMP 陷阱和插件。Internet 上的 SNMP 陷阱服务接收本地或远程 SNMP 代理生成的陷阱消息，然后将这些消息转发给控制台上运行的 SNMP 管理程序。为代理配置了 SNMP 陷阱服务后，如果发生任何特定的事件，都将生成陷阱消息。这些消息被发送到陷阱目标。有些 IDS 可以与网络管理工具一起使用，它能使用网络管理基础设施来传送在网络管理控制台显示的告警和警报信息，它依附简单网络管理协议(SNMP)的消息或陷阱作为一个告警选项。

8.2.2 主动响应入侵检测系统

主动响应系统可以分为两类：对被攻击系统实施控制的系统和对攻击系统实施控制的系统。对攻击系统实施控制比较困难，主要是对被攻击系统实施控制。主动响应有两种形式，一种是由用户驱动的，一种是由系统本身自动执行的。对入侵者采取反击行动，修正系统环境和收集尽可能多的信息是主动响应的基本手段。

对入侵者采取反击行动是最主动的一种响应方式。警告攻击者、跟踪攻击者、断开危险连接和对攻击者进行攻击是最严厉的一种主动反击手段，但这种响应方式具有一定的风险。温和的方式是记录安全事件、产生报警信息、记录附加日志和激活附加入侵检测工具等，这种方式显得不够主动。介于严厉与温和之间的手段有隔离入侵者 IP、禁止被攻击对象的特定端口和服务以及隔离被攻击对象等，这种方式可行性较好。

修正系统环境与直接采取反击相比，主动性要差一些，然而当与提供调查支持的响应结合在一起时，却往往是一种更好的响应方案。这类响应也许通过增加敏感程度改变分析引擎的操作特征，通过插入规则改变专家系统，即通过这些规则提高对某些攻击的怀疑水平，或增加监视范围以更好地收集信息。这种策略其实是一种反馈机制，即目前系统处理过程的输出将用来调整和优化下一个处理过程。

收集额外信息是主动响应的第三种方法。当被保护的系统非常重要并且管理员想对系统的配置进行改进时，这种方法特别有用。这种方法类似于蜜罐技术，以这种方式收集的信息对于管理员对网络安全情况进行分析，以及改进系统的安全防护是很有效的。

8.2.3 应急响应组

所谓应急响应即"Incident Response"或"Emergency Response"，通常指一个组织为了应对各种意外事件的发生所做的准备以及在事件发生后所采取的措施。应急响应的开始是因为有"事件"发生。所谓"事件"或"安全事件"指的是那些影响计算机系统和网络安全的不当行为。网络安全事件造成的损失往往是巨大的，而且往往是在很短的时间内造成的。因此，应对网络事件的关键是速度与效率。应急响应技术在"事件"方面的内容包括事件分类、事件描述和事件报告等。

应急响应组是专门处理安全事件的组织，常用的名字是 CERT 或 CSIRT，指 Computer Emergency Response Team（计算机紧急响应组）或 Computer Security Incident Response Team（计算机安全应急响应组），它们的成员需要在协调能力和专业知识两方面都达到一定的水平。这是因为网络安全事件往往牵连很广，不只是一个部门甚至不只是一个领域；安全事件现在越来越复杂，对网络安全和计算机系统了解不够或者经验不足都无法保证工作效率。

应急响应组织都必须明确自己的政策、功能与角色，明确服务的对象并与之保持联系，在此基础上制订适合的政策，建立事件报告渠道、应急响应流程和应急通信机制。但不同的应急响应组织由于其角色定位不同，其运行也有很大不同。主要存在两类组织，一种是企业或单位内部的应急响应小组，只为本单位服务，人员主要由网络管理员和各部门中接受了训练的人员组成；另一种是面向区域的应急响应组织，比如 CERT/CC（美）和 DFN-CERT（德）、AusCERT（澳）等，它们定位于区域甚至整个 Internet；不但对安全事件提供响应报

务,同时进行安全技术的研究、安全形势的分析或者发布预警信息等,它们的人员一般由固定全职人员(承担技术研究)和兼职人员组成。但无论哪种情况,应急响应组至少都有协调能力、专业知识、响应效率、前期主动防御能力、适应需求、联络能力和处理制度障碍能力上的优势。

应急响应组最难的重要问题是人员问题,既包括管理人员也包括技术人员。两者的重要性与两者合作的重要性不言自明。管理人员的管理能力是应急响应组必需的,他并不需要在技术上有多强。一个响应组的管理要比它拥有的技术专家重要,这似乎与今天普遍重视技术实力的情况相反。但是事实上确实如此,因为从经济角度考虑,响应组只需要维护一个专家的联系清单就足够了;技术专家也可能更愿意把精力投入到他们认为更重要的事情。而技术上日常的工作完全可以交给普通的技术人员。

人员问题的另一个方面是沟通与合作,不只包括响应组内部的沟通与合作,还包括与受害者以及其他相关部门(如法律机构、媒体等)的沟通与合作。这在响应组的运转尤其是响应过程中对效率问题有很大的影响。

8.2.4　应急响应计划

应急响应计划(Emergency Response Plan,ERP)是指组织为了应对突发/重大信息安全事件而编制的,对包括信息系统运行在内的业务运行进行维持或恢复的策略和规程。

应急响应计划与应急响应是相互补充与促进的关系。首先,应急响应计划为信息安全事件发生后的应急响应提供了指导策略和规程,否则,应急响应计划将陷入混乱,而毫无章法的应急响应有可能造成比信息安全事件本身更大的损失。其次,应急响应可能发现事前应急响应计划的不足,从而吸取教训,进一步完善应急响应计划。

制定信息安全应急响应计划是一个周而复始、持续改进的过程,包含以下几个阶段。

1. 应急响应计划的编制准备

应急响应计划的编制准备是编制应急响应计划文档的前期工作,由风险评估、业务影响分析和制定应急响应策略三个方面组成。

风险评估的目的是标识信息系统的资产价值,识别信息系统面临自然地和人为地威胁,识别信息系统的脆弱性,分析各种威胁发生的可能性。风险评估是业务影响分析的基础。

业务影响分析(BIA)是在风险评估的基础上分析各种信息安全事件发生时对业务功能可能产生的影响,进而确定应急响应的恢复目标,因此它是应急响应计划制定前期的一项重要任务。

BIA 的目的是将特定的系统组件与其提供的关键服务联系起来,并基于这些信息了解系统部件中断所产生的影响的特点,通过 BIA 估计业务停顿随时间而造成的损失,进而确定对企业而言比较合适的回复时间目标。BIA 的一个重要目的是确定应急响应的恢复目标,应急响应恢复目标包括:①关键业务功能及恢复的优先顺序;②恢复时间范围,即恢复时间目标(RTO)和恢复点目标(RPO)的范围。

应急响应策略提供了在业务中断、系统死机、网络瘫痪等突发或重大信息安全事件发生后快速有效地恢复信息系统运行的方法。这些策略应涉及在 BIA 中确定的应急响应的恢复目标。制定应急响应策略主要需要考虑"系统恢复能力等级划分"、"系统恢复资源的要求"和"费用"三个因素。组织进行成本效益分析,以确定最佳应急响应策略。

2. 编制应急响应计划文档

编制信息安全应急响应计划文档是应急响应规划过程中的关键一步。应急响应计划文档应包含总则、角色及职责、预防和预警机制、应急响应流程、应急响应保障措施、附件六个基本要素。

总则提供了重要的背景或相关信息，使应急响应计划更容易理解、实施和维护。通常这部分包括编制目的、编制依据、适用范围、工作原则等。

组织应结合本单位日常机构建立信息安全应急响应的工作机构，并明确其职责。应急响应的工作机构由管理、业务、技术和行政后勤等人员组成。一般来说，按角色可划分为三个功能小组：应急响应领导小组、应急响应实施小组和应急响应日常运行小组等。实际中，可以不必专门成立对应的功能小组，组织可以根据自身情况由具体的某个或某几个部门或部门中的某几个人担当其中的一个或几个角色。

预防和预警机制是一种防御性的方法，在可行和比较划算后，防御性方法要比信息安全事件发生后进行应急要更好。有很多防御性控制措施可供选择，它依赖于信息系统的类型和配置。

应急响应流程描述并规定了信息安全事件发生后应采取的工作流程和相应条款，目的是保证应急响应能够有组织地执行，从而最大程度地保证应急响应的有效性。

应急响应保障措施是信息安全应急响应计划的主要组成部分，是保证信息安全事件发生后能够快速有效地实施应急响应计划的关键要素。考虑到各个组织的性质和需求可能存在很大的差异，标准中描述的具体内容是可选择的，也可以做适当调整，但人力保障、物质保障和技术保障这三大方面是必要的。

3. 应急响应计划测试、培训、演练和维护

为了检验应急响应计划的有效性，同时使相关人员了解信息安全应急响应计划的目标和流程，熟悉应急响应的操作规程，组织还应按照一定的要求对应急响应计划进行测试、培训、演练和维护。

计划的测试是有效的应急响应计划的关键要素。测试能确定和解决计划的缺陷，并协助评估和应急人员快速有效实施应急响应计划。每一个信息安全应急响应计划要素都应得到测试，以确保各个恢复规程的正确性和计划整体的有效性。

计划的培训是对测试的补充。培训至少每年举办一次，拥有计划规定职责的新雇员应该在被雇佣后接受短期培训。和应急响应计划相关的人员所接受的培训最终使得他们能够无需实际文档的协助就能够执行相应的恢复规程。这在信息安全事件影响造成的最初几个小时里未获得书面或电子版本的应急响应计划的情况具有非常重要的意义。

计划的演练对小组成员来说是有益的，这样可以使他们做好精神准备并有时间对工作负荷进行优化调整。

由于业务需要的转移、技术的更新或新的内外政策会造成信息系统的频繁变化，应急响应计划的定期检查和更新是至关重要的，应作为机构变化管理过程的一部分以确保新的信息能够被添加进来，应急响应措施能够根据需要被修订。计划应该至少每年进行一次针对正确性和完整性的检查，在计划的任何部分发生重大变化时也应该进行，这是一项基本要求。某些部分应该得到频繁的检查，如联络清单。根据信息系统类型和重要程度的不同，对计划内容和规程的评估可能会更加频繁。

8.3　入侵检测的分析技术

1. 异常检测

异常检测技术也称为基于行为的检测技术，是根据用户的行为和系统资源的使用状况判断是否存在网络入侵。异常检测基于一个假设，即入侵行为存在于偏离系统正常使用的事件中，异常检测就是要找出偏离正常行为的事件，并从中发现入侵。通过将过去观察到的正常行为与受到攻击时的行为加以比较，根据使用者的异常行为或资源的异常使用状况来判断是否发生入侵活动，其原则是任何与已知行为模型不符合的行为都认为是入侵行为。

异常检测的假设是入侵者活动异常于正常主体的活动。这种活动存在 4 种可能：入侵性而非异常、非入侵性且异常、非入侵性且非异常、入侵且异常。异常检测一般先建立用户正常行为的模型，再将实际观察到的行为与之相比较，检测与正常行为偏差较大的行为。因此，如果能够建立系统正常运行的轨迹，那么理论上可以把所有与正常轨迹不同的系统状态视为可疑企图。模型如图 8-5 所示。

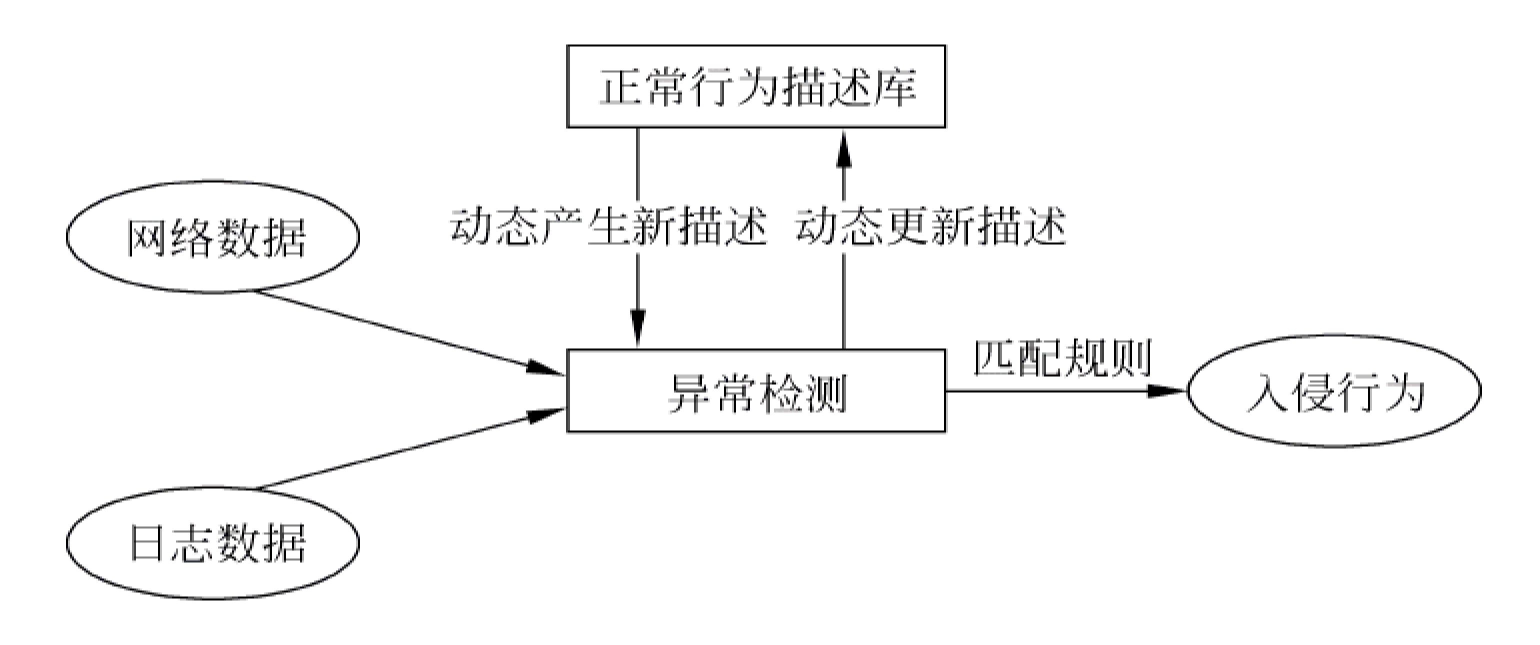


图 8-5　异常检测模型

根据这一理念，异常检测方法通过异常监视器观察主体活动，建立主体正常活动的“活动轮廓”，将当前主体的活动状况与已存储的“活动轮廓”相比较，检测“入侵”行为。

异常检测的优点是，与系统相对无关，通用性比较强，易于实现，易于自动更新，可以发现未知的入侵行为，同时有一定的学习能力。但单纯的统计异常检测方法对时间发生的次序不够敏感，误报、虚报率较高，对没有统计特征的攻击方法难以检测。

异常检测的难题在于如何建立系统正常行为的“活动轮廓”以及如何设计统计算法，基于异常的入侵检测系统通过构造不同的异常模型来实现不同的检测方法，使用观测到的一组测量值偏离度来预测用户行为的变化而做出决策判断。目前基于异常的入侵检测系统主要采用的技术有统计分析、贝叶斯推理、神经网络和数据挖掘等方法。

2. 误用检测

误用检测技术也称为基于知识的检测或基于特征的检测。误用检测假设所有入侵的行为或手段及其变种都能表达为一种模式或特征。误用检测技术通过对已知的入侵行为和手段进行分析，提取检测特征，构建攻击模式或攻击签名。检测时，主要判别主机或者网络中所搜集到的数据特征是否匹配所收集的特征库中的一种，以此判断是否有入侵行为。

基于误用检测的入侵检测技术通过收集入侵攻击和系统缺陷的相关知识来构成入侵检测系统中的知识库,然后利用这些知识寻找那些企图利用这些系统缺陷的攻击行为。也就是说基于误用检测的入侵检测技术是通过检测那些与已知的入侵行为模式类似的行为或间接地违背系统安全规则的行为来判断是否有入侵活动。系统中的任何不能确认是攻击的行为都被认为是系统的正常行为。

基于误用的入侵检测技术具有良好的精确度,误报率较低,但其检测依赖于对入侵攻击和系统缺陷相关知识的不断更新和补充,因此不能检测未知的入侵行为。误用检测的关键是如何表达入侵行为、构建攻击模式,把真正的入侵与正常行为区分开,以及检测过程中的推理模型。在具体实现上,基于误用检测的IDS知识在表示入侵模式的方式以及在系统的审计入侵模式的机制上有所不同。误用检测的主要实现技术有专家系统、状态转换分析、模式匹配等。

8.4 入侵检测系统的结构与部署

1. 入侵检测系统的结构

为了提高IDS产品、组件以及与其他安全产品之间的互操作性,美国国防高级研究计划署与Internet工程任务组的入侵检测工程组(IDWG)发起制定了一系列建议草案,从体系结构、API、通信机制、语言格式等方面规范IDS的标准,提出了通用入侵检测框架(Common Intrusion Detection Framework,CIDF)。

CIDF阐述了一个入侵检测系统的通用模型,其结构如图8-6所示。它将一个入侵检测系统分为以下组件:事件产生器(event generators)、事件分析器(event analyzers)、响应单元(response units)、事件数据库(event databases)。

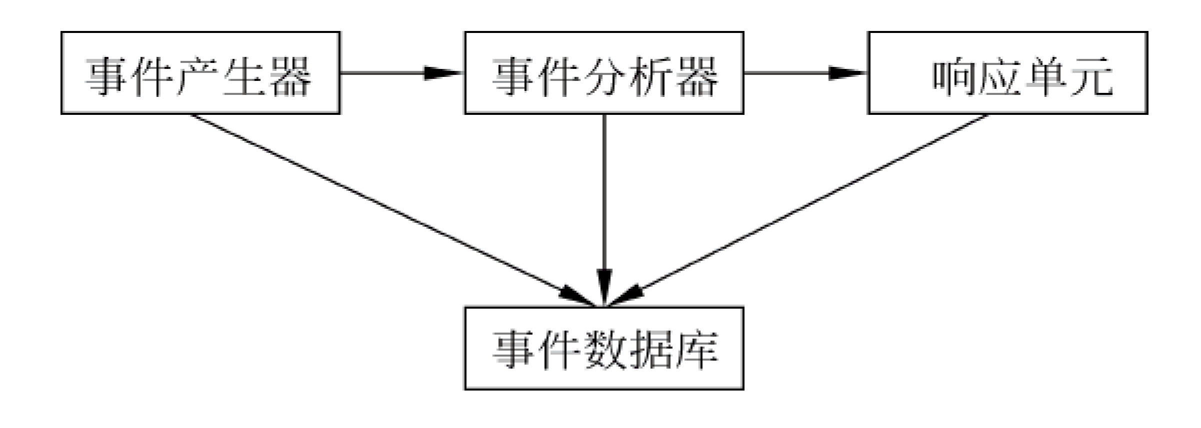


图8-6 CIDF体系结构

CIDF将需要分析的信息称为事件,它既可以是网络中的数据包,也可以是从系统日志等其他途径得到的信息。事件产生器一般分布于网络的各个敏感关节点处或某个主机上,它从整个网络环境中获得事件,并向系统的其他部分提供事件。

事件分析器分析所得到的数据并提取其有效特征(如网络流、数据包报头、数据包序列、日志文件参数、审计数据间的关系等)以产生分析结果,判断是否属于攻击事件。事件分析器性能的优劣决定了入侵检测系统性能的高低。

一旦检测到攻击,事件分析器会通知响应单元,后者做出反应,如终止进程、切断连接、改变防火墙规则、修改文件属性等。事件数据库用于存储和管理事件数据,它可以是复杂的数据库,也可以是简单的文本文件。

2. 分布式的入侵检测系统

随着网络系统结构的复杂化和大型化，出现了许多新的入侵检测问题，下面列举几个。

首先，系统的弱点或漏洞分散在网络中的各个主机上，这些弱点问题可能被入侵者一起用来攻击网络，而仅依靠一个主机或网络入侵检测系统难以发现入侵行为；

第二，网络入侵行为不再是单一的行为，而是表现出相互协作的入侵特点，例如，分布式拒绝服务攻击；

第三，入侵检测系统所依靠的数据来源分散化，使得收集原始的检测数据变得比较困难，例如交换型网络使得监听网络数据包受到限制；

第四，网络速度传输加快，网络的流量大，原始数据的集中处理方式往往造成检测瓶颈，从而导致漏检。

因此，产生了分布式入侵检测系统，分布式 IDS 的目标是既能检测网络入侵行为，又能检测主机的入侵行为。系统通常由数据采集模块，通信传输模块、入侵检测分析模块、响应处理模块、管理中心模块和安全知识库组成。这些模块可根据不同情况进行组合，例如数据采集模块和通信模块组合产生出的新模块就能完成数据采集和传输两种任务。所有的这些模块组合起来就变成了一个入侵检测系统。需要指出的是，模块按照网络配置情况和检测的需要可以安装在单独的一台主机上或者分散在网络的不同位置，甚至一些模块本身就能够单独地检测本地的入侵，同时将入侵检测的局部结果信息提供给入侵检测管理中心。

1991 年，加州大学戴维斯分校的研究人员提出了分布式入侵检测系统的概念，将以往基于主机和网络的入侵检测系统结合起来，给出了一种分布式入侵检测系统的体系结构。如图 8-7 所示，整个系统包括三个大部分，位于每台监控主机上的入侵传感器，局域网上的局域网管理器和中央数据处理单元。主机上的入侵传感器和局域网管理器分别从主机和局域网上采集有用数据，然后将数据送至中央数据处理单元做全局的入侵检测。这个体系结构侧重于网络-用户识别问题的解决，即通过跟踪用户在网络上的活动情况，计算用户操作的相关性来判断是否有入侵行为发生。这个体系结构运用范围较小，而且由于模型过于简单，无法检测复杂的入侵行为。要使系统具有更强的检测能力，需要进一步完善系统体系结构和检测模型。

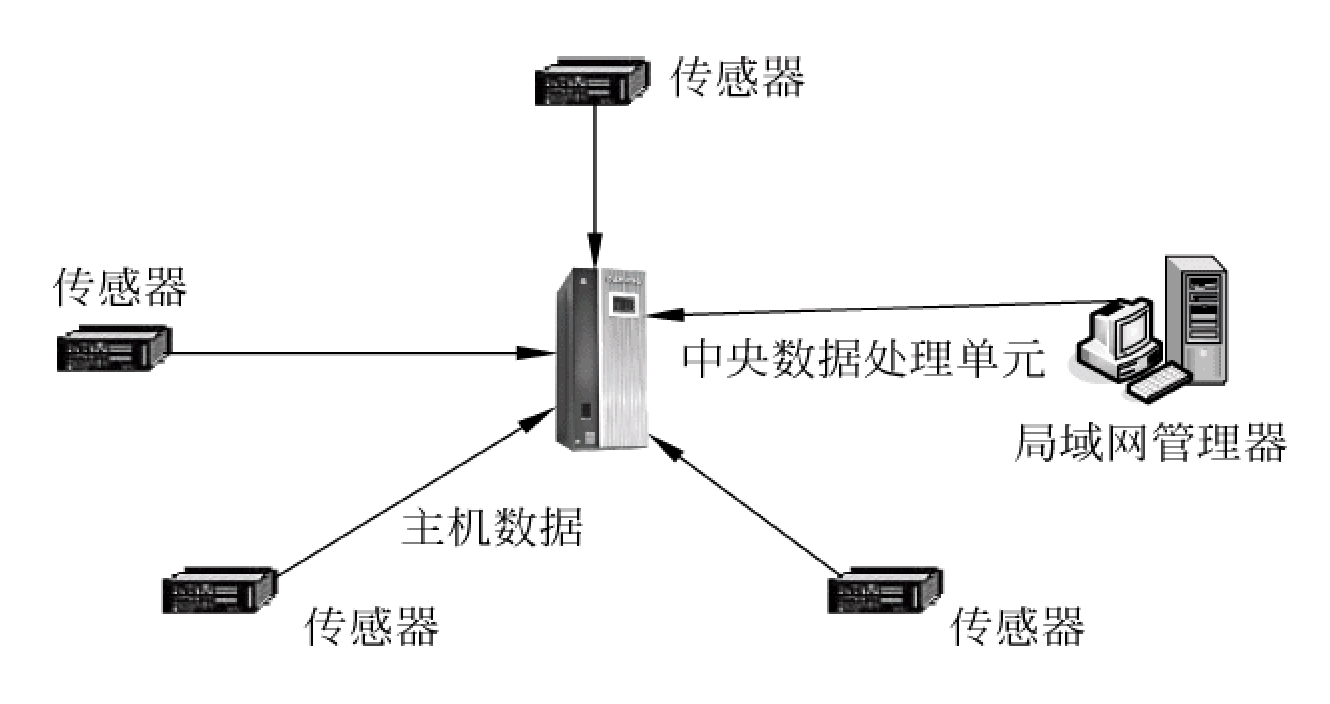


图 8-7　早期的分布式入侵检测模型

在入侵检测的体系结构中，哥伦比亚大学和北卡罗来州立大学的研究人员在 2000 年提出了基于数据挖掘的模型建立方法。该方法将各个传感器搜集到的数据放入数据库，利用数据挖掘的方法提取出有效信息，自适应地建立检测模型，该系统的体系结构如图 8-8 所示。

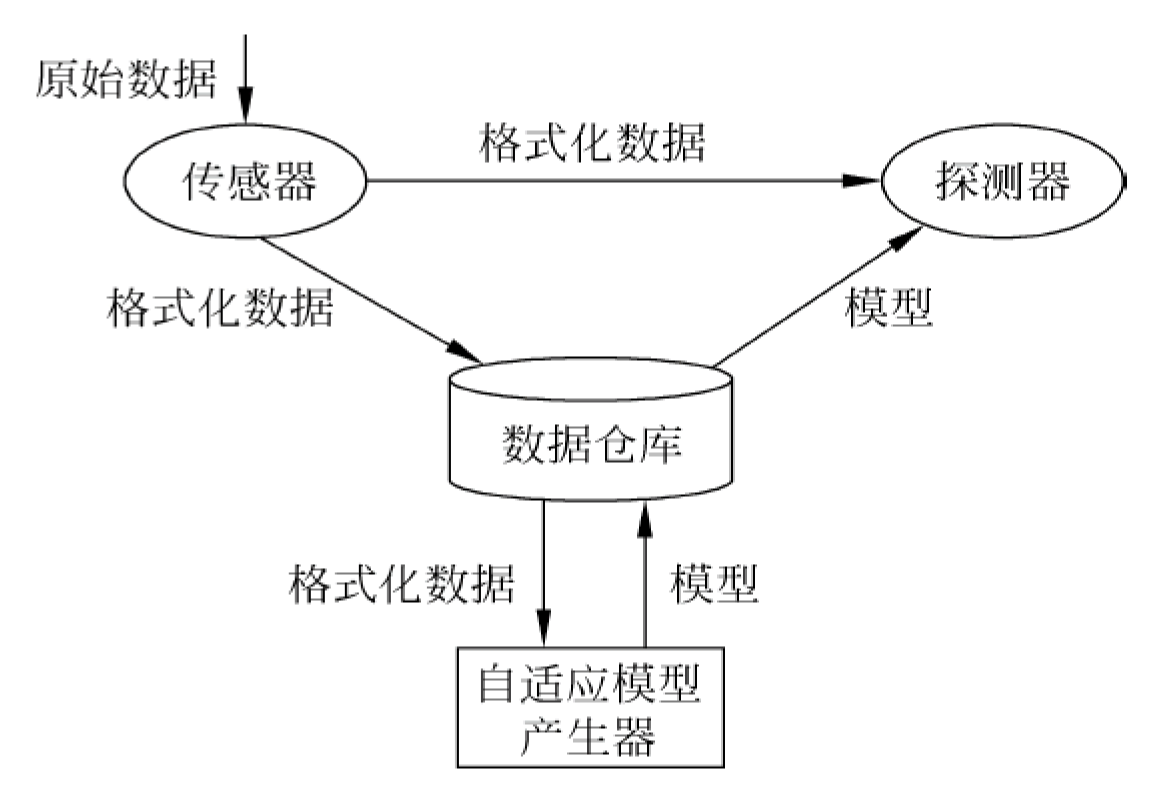


图 8-8　改进的入侵检测模型示意图

该系统由传感器、探测器、数据仓库和自适应模型产生器四个主要部件构成。传感器接收从网络或者主机来的原始数据，从原始数据中提取事件，并按照预定的格式将数据发送给检测器和数据仓库。检测器接收从传感器发送来的数据，并将这些数据与已有的入侵模型进行匹配，判断是否有入侵行为发生。数据仓库则存放格式化的数据和已有的模型。模型产生器从数据仓库中获取数据，按照这些数据自适应地建立或调整检测模型，并将模型存放到数据仓库中。该模型的特点是，各模块之间采用统一的数据格式通信，使得获取原始数据有很大的灵活性。原始数据可以来自主机，也可以来自网络。其次，模块独立，且模块具有自适应性。但是，由于格式化的数据都存放在数据仓库中，并有单独的模型产生器读取数据，建立、修改模型，所以扩展性较差。

8.5　入侵检测系统 Snort

8.5.1　Snort 概述

Snort 是一款免费的基于 Libpcap 的轻量级网络入侵检测系统。它具有易于配置、检测效率高的特点。Snort 具有实时的流量分析和 IP 数据包日志分析能力，能够对协议进行分析或对内容进行实时搜索。Snort 能够检测不同的攻击行为，如缓冲区溢出攻击、端口扫描、拒绝服务攻击等，并实时地报警。

Snort 的安装和配置有两种选择，一种是只安装 Snort，另一种是安装 Snort 的同时安装其他的工具软件，从而搭建一个完整的 Snort 系统。在第一种的安装方式下，Snort 会将捕捉到的入侵检测数据以文本或二进制的形式保存在文件中，同时将生成的警告信息发送到 SNMP 管理器，这种方式使得用户面对大量的日志和警告信息。第二种方式，用户不但可以将入侵检测数据保存到数据库中，而且可以利用工具对入侵检测的数据进行分析。

8.5.2　搭建 Windows 入侵检测平台

下面介绍搭建 Windows 入侵检测平台用到的软件及其版本。

(1) pache_2.0.63_win32(Windows 版本的 apache Web 服务器)，用作 Web 服务器。

(2) PHP_4.3.2_win32,作为 Web 服务器和 MySQL 数据库之间的接口。

(3) snort_2.0.0_win32 ,入侵检测软件 Snort。

(4) mysql_4.0.23_win32 ,用来存放 Snort 产生的警告信息,也可以使用 Oracle 等数据库。

(5) adodb465, ACID 用来连接 MySQL 数据库。

(6) acid_0.9.6b23 是一个用 PHP 编写的软件包,用来查看和分析 Snort 产生的入侵检测数据。

(7) jpgraph_1.26 用于生成 Snort 产生的入侵检测数据的统计图形。

(8) winpcap_3.2, 用于信息包捕获和网络分析,安装 Snort 所必需的软件包。

下面介绍其具体步骤。

1. 安装 apache_2.0.63 For Windows

(1) 双击 apache_2.0.63 安装包,安装程序默认安装至 C:\Program Files\Apache Group 目录。

如果主机已经安装了 IIS,并启动了 WebServer,由于 IIS WebServer 与 Apache WebServer 都默认在 TCP80 端口监听,所以会和 Apache WebServer 冲突。解决方法是,停止 IIS 工作或修改 Apache WebServer 的默认端口。修改 Apache WebServer 的监听端口为 18080,打开 Apache 的配置文件 C:\Program Files\Apache Group\Apache2\conf\httpd.conf,将 Listen 80 改为 Listen 108080,如图 8-9 所示。

```
# Change this to Listen on specific IP addresses as shown below to
# prevent Apache from glomming onto all bound IP addresses (0.0.0.0)
#
#Listen 12.34.56.78:80
Listen 18080
```

图 8-9　修改 Apache WebServer 的监听端口

(2) 安装 Apache 为服务方式运行:运行命令 C:\Program Files\Apache Group\Apache2\bin\apache-k-install,而且 apache_2.0.63 提供了图形界面支持,方便启动和关闭 Apache 服务,如图 8-10 所示。

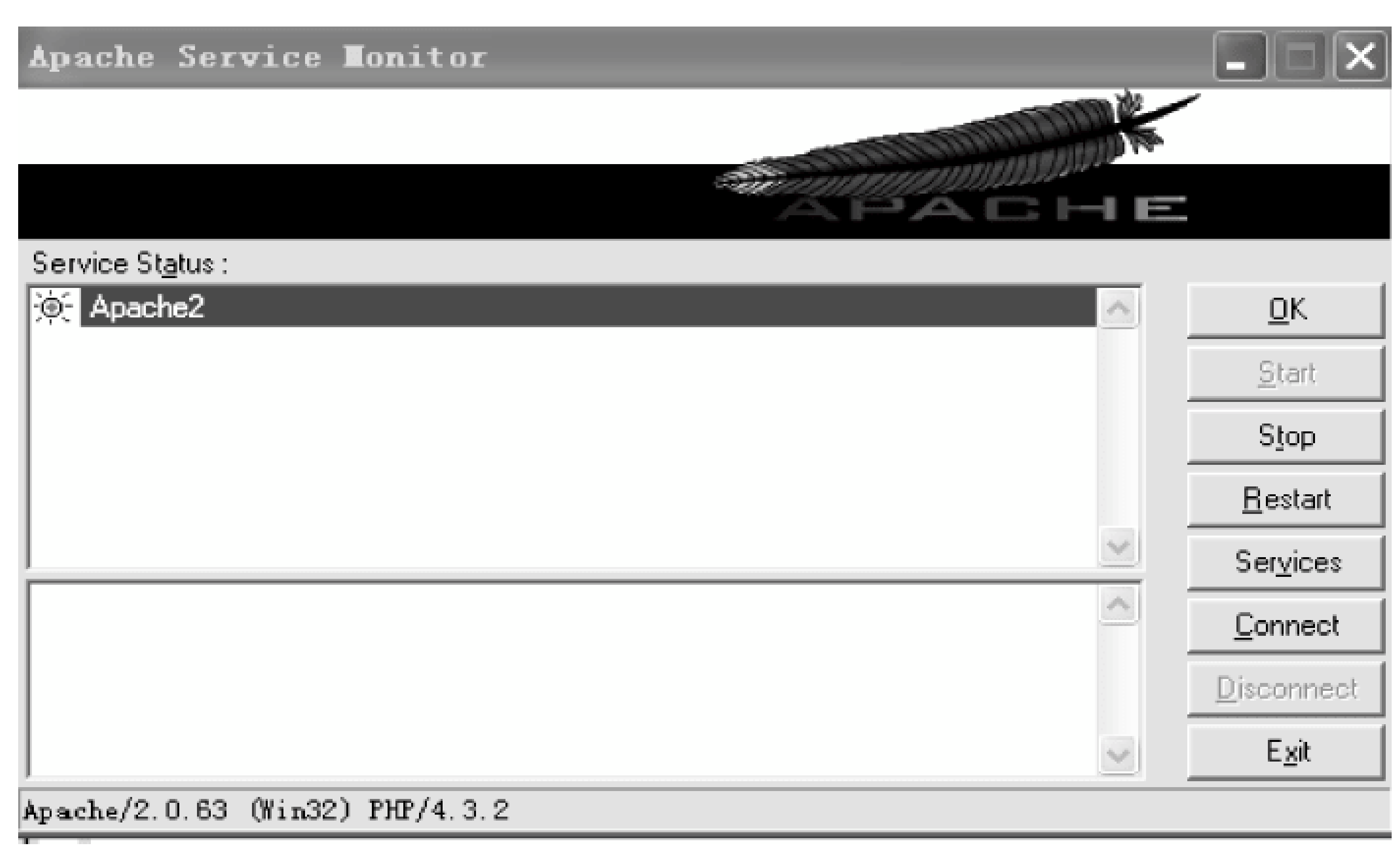


图 8-10　Apache 服务

(3) 检验 Apache 安装成功：在浏览器中输入 http：//127.0.0.1：18080，若能看到图 8-11 所示的信息，则说明 Apache 安装成功。

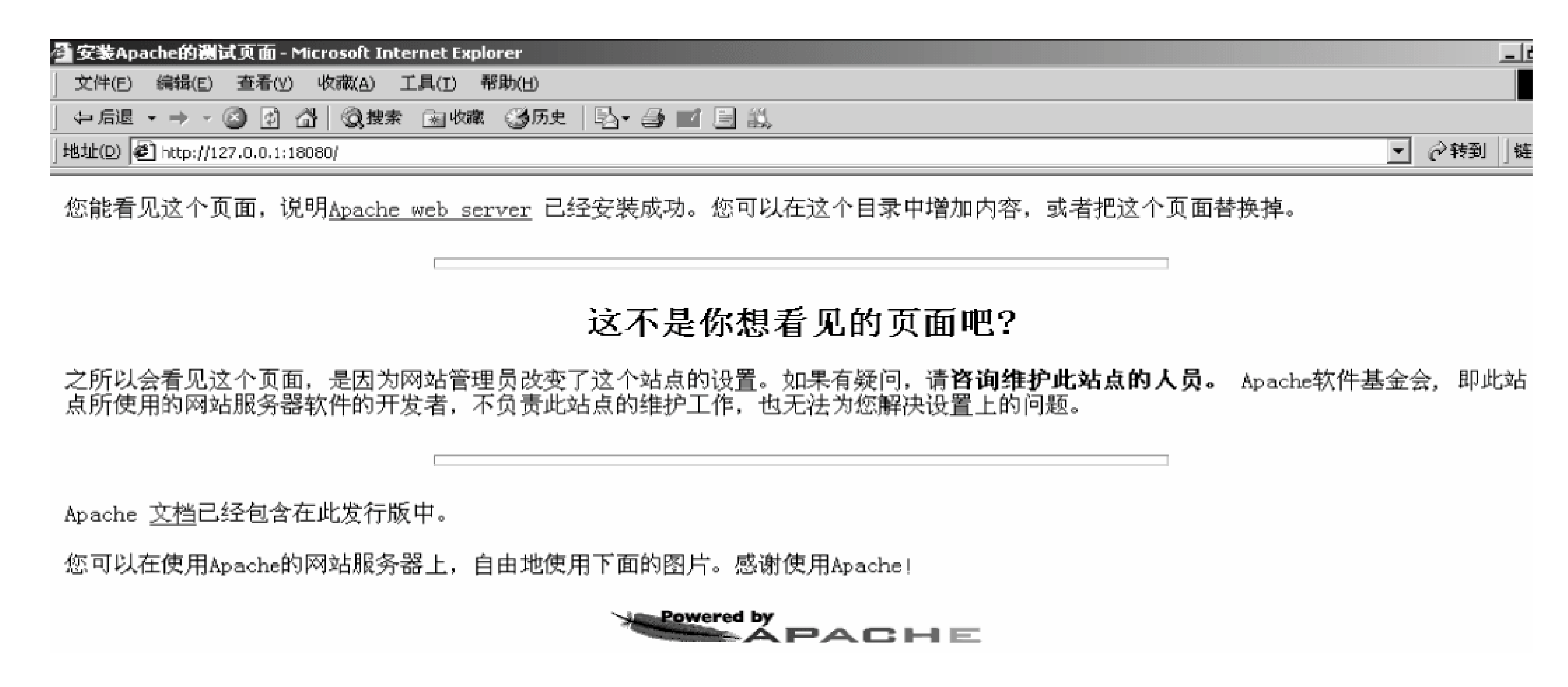


图 8-11　检验 Apache 安装成功

(4) 添加 Apache 对 PHP 的支持。

① 解压缩 php-4.3.2-win32.zip 至 c：\php；

② 复制 php4ts.dll 至操作系统目录 c：\windows\system32；

③ 将 c：\php\php.ini-dist 改名为 php.ini，并且复制到操作系统目录 c：\windows\system32；

④ 修改 c：\php\php.ini，将文件 php.ini 中的 extention＝php_gb2.dll 的注释“；”去掉，以添加对 gb 图形库的支持，如图 8-12 所示。

(5) 启动 Apache 服务：运行 net start apache2 命令，如图 8-13 所示。

```
;extension = php_fdf.dll
;extension = php_filepro.dll
extension = php_gd2.dll
;extension = php gettext.dll
```

图 8-12　添加 Apache 对 PHP 的支持



图 8-13　启动 Apache 服务

(6) 测试 Apache 对 php 的支持已经成功安装：在 C：\Program Files\Apache Group\Apache2\htdocs 目录下建立 test.php 文件，文件的内容如图 8-14 所示。

在浏览器中输入 http：//127.0.0.1：18080/test.php，如图 8-15 所示，说明成功安装了 Apache 对 php 的支持。

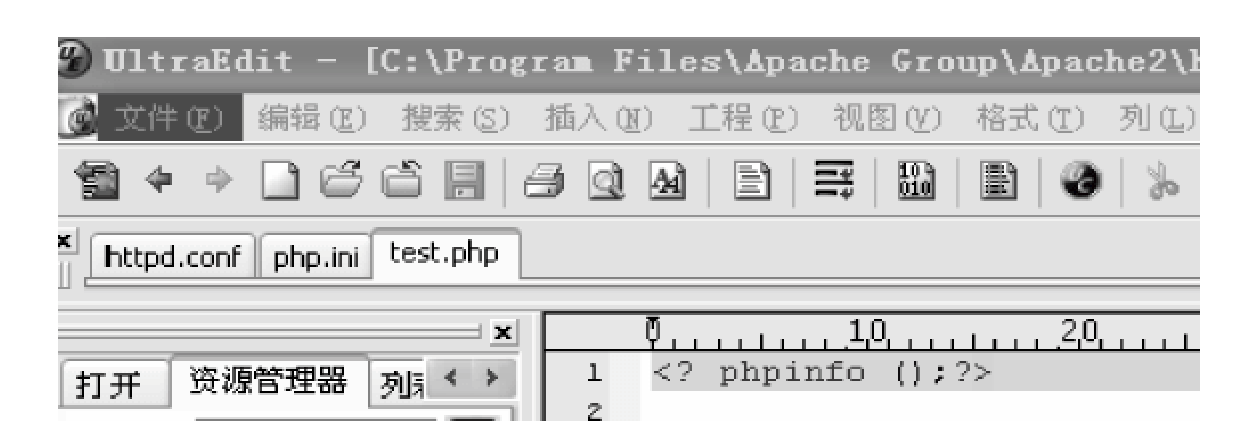


图 8-14　测试 Apache 对 php 的支持已经成功安装

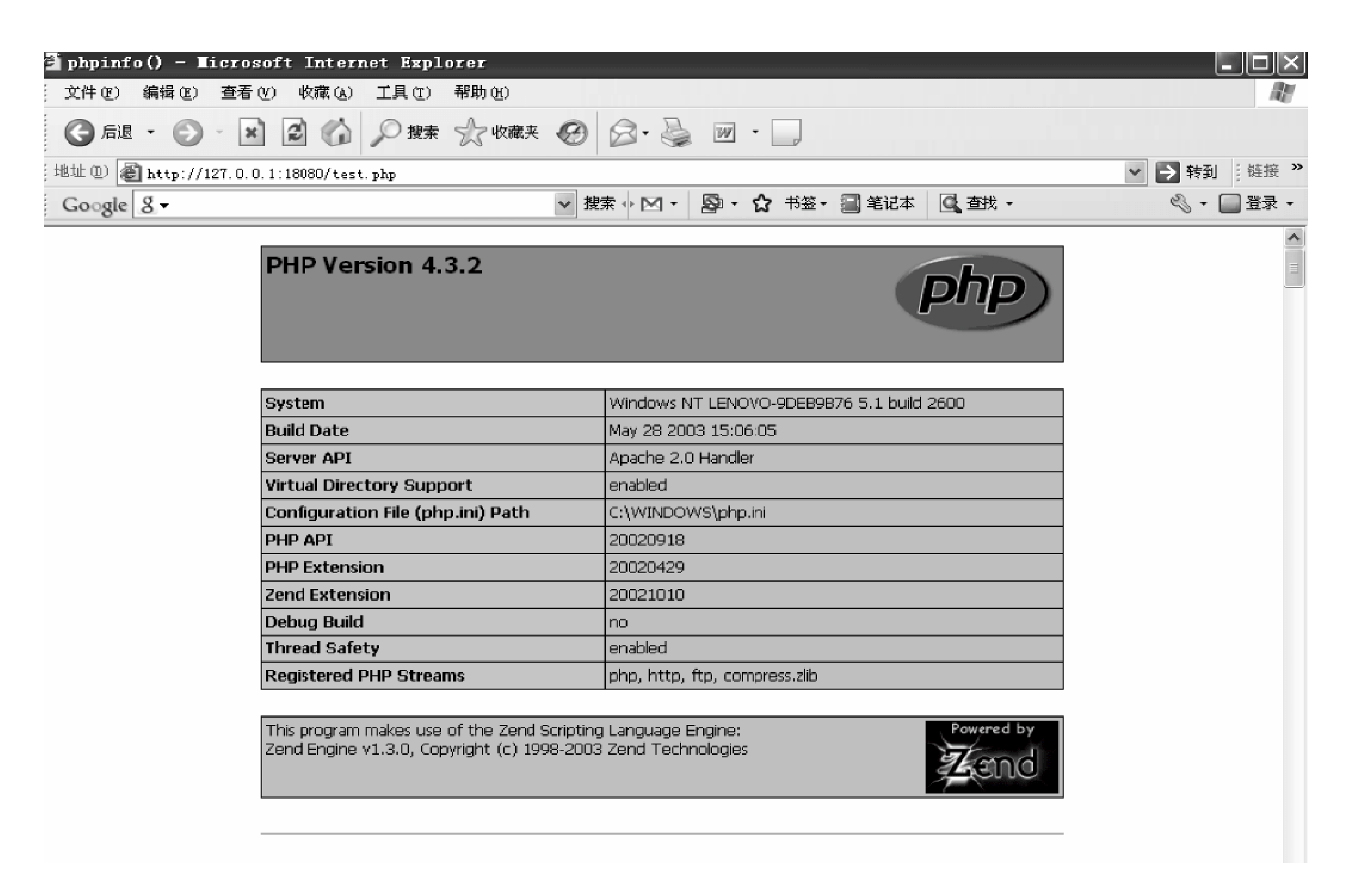


图 8-15　成功安装 Apache 对 php 的支持

2. 安装 Snort 2.0.0 与 winpcap

直接双击 Snort 2.0.0.exe 文件，将默认安装 Snort 到 C：\snort(Snort 的新版本的规则库现在已经收费)。

3. 安装、配置 MySQL 数据库

(1) 默认安装 mySQL 4.0.23 for windows 至 C：\mysql。

(2) 安装 MySQL 为服务方式运行。

运行命令：C：\mysql\bin\mysqld-nt-install，如图 8-16 所示。

(3) 启动 MySQL 服务。

运行命令：net start mysql，如图 8-17 所示。




图 8-16　安装 MySQL



图 8-17　启动 MySQL 服务

(4) 为 root 用户配置口令。

首先，以 root 用户登录数据库 mysql(运行命令 mysql-u root-p，出现 Enter password：提示符直接回车，初始时默认 root 用户无密码登录数据库)，运行命令 set password for "root"@"localhost" ＝ password('123')；，如图 8-18 所示。

需要注意的是，需要输入"；"，MySQL 才会编译此命令。

```
C:\mysql\bin>mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2 to server version: 4.0.23-nt

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> set password for "root"@"localhost" = password('123');
Query OK, 0 rows affected (0.00 sec)

mysql> _
```

图 8-18　为 root 用户配置口令

（5）建立 Snort 运行必须用到的 Snort 数据库和 Snort_archive 数据库。

① 运行命令 mysql>create database snort；

mysql>create database snort_archive；，如图 8-19 所示。

此时可以利用命令 show databases；查看两个数据库是否已建立成功：

mysql>show database；，如图 8-20 所示。

```
mysql> create database snort;
Query OK, 1 row affected (0.01 sec)

mysql> create database snort_archive;
Query OK, 1 row affected (0.00 sec)
```

图 8-19　建立 Snort 数据库和 Snort_archive 数据库



```
mysql> show databases;
+---------------+
| Database      |
+---------------+
| mysql         |
| snort         |
| snort_archive |
| test          |
+---------------+
4 rows in set (0.02 sec)
```

图 8-20　查看两个数据库是否已建立成功

② 建立 snort 运行所必需的数据表

首先输入 quit 退出 mysql，在出现的提示符后输入命令：

c:\mysql\bin> mysql -D snort -u root -p < c:\snort\contrib\create_mysql

c：\mysql\bin>mysql -D snort_archive -u root -p <c：\snort\contrib\create_mysql，如图 8-21 所示。

```
C:\mysql\bin>mysql -D snort -u root -p < c:\snort\contrib\create_mysql
Enter password: ***

C:\mysql\bin>mysql -D snort_archive -u root -p < c:\snort\contrib\create_mysql
Enter password: ***

C:\mysql\bin>
```

图 8-21　建立 snort 运行所必需的数据表

（6）建立 acid 和 Snort 用户：运行 mysql -u root -p，以 root 用户身份登录 MySQL 数据库，当然此时需要输入 password：123，输入下面的命令建立 acid 和 snort 用户，如图 8-22 所示。

```
mysql> grant usage on *.* to "acid"@"localhost" identified by "acid123";
mysql> grant usage on *.* to "snort"@"localhost" identified by "snort123";
```

```
C:\mysql\bin>mysql -u root -p
Enter password: ***
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 5 to server version: 4.0.23-nt

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> grant usage on *.* to "acid"@"localhost" identified by "acid123";
Query OK, 0 rows affected (0.06 sec)

mysql> grant usage on *.* to "snort"@"localhost" identified by "snort123";
Query OK, 0 rows affected (0.00 sec)
```

图 8-22　建立 acid 和 Snort 用户

(7) 赋给用于 acid 和 snort 相应的权限。

```
mysql > grant select, insert, update, delete, create, alter on snort . * to "acid"@"localhost";
mysql > grant select, insert on snort . * to "snort"@"localhost";
mysql > grant select, insert, update, delete, create, alter on snort . * to "acid"@"localhost";
```

如图 8-23 所示：

```
mysql> grant select,insert,update,delete,create,alter on snort.* to "acid"@"loca
host";
Query OK, 0 rows affected (0.00 sec)

mysql> grant select,insert on snort.* to "snort"@"locahost";
Query OK, 0 rows affected (0.00 sec)

mysql> grant select,insert,update,delete,create,alter on snort.* to "acid"@"loca
host";
Query OK, 0 rows affected (0.00 sec)
```

图 8-23　赋给用于 acid 和 snort 相应的权限

4. 安装 adodb

解压缩 adodb464 至 C：\php\adodb 目录下。

5. 安装 acid

(1) 解压缩 acid-0. 9. 6b23. tar. gz 至 C：\Program Files\Apache Group\Apache2\htdocs\acid 目录下。

(2) 修改 acid 的配置文件 acid_config. php：

```
$DBlib_path = "c:\php5\adodb";
$alert_dbname = "snort";
$alert_host = "localhost";
$alert_port = "3306";
$alert_user = "acid";
$alert_password = "acid123";
/* Archive DB connection parameters */
$archive_dbname = "snort_archive";
$archive_host = "localhost";
$archive_port = "3306";
$archive_user = "acid";
$archive_password = "acid123";
$ChartLib_path = "c:\php5\jpgraph\src";
```

(3) 在浏览器下输入 http：//127. 0. 0. 1：18080/acid/acid_db_setup. php，出现图 8-24。

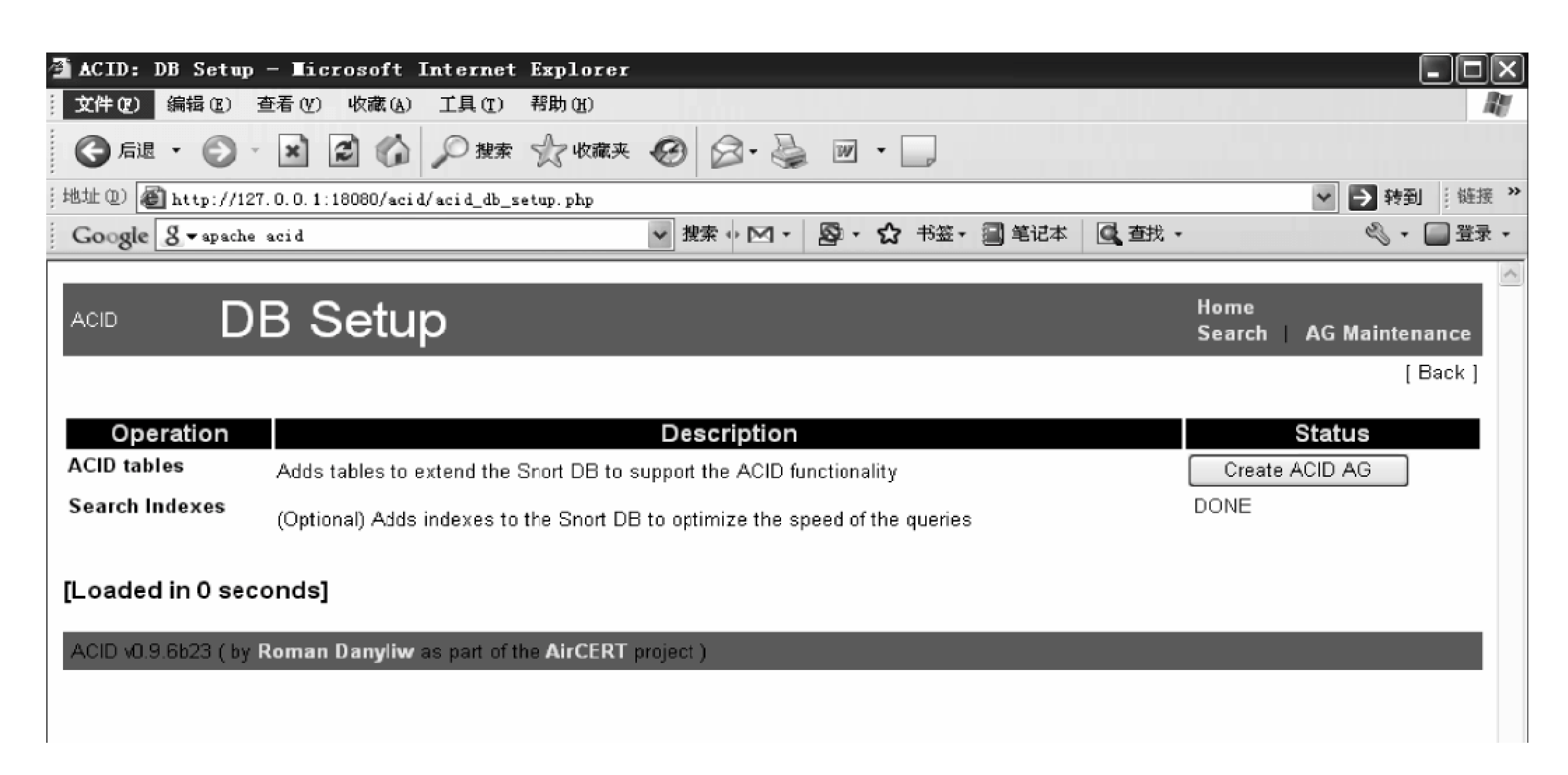


图 8-24　输入 http://127.0.0.1:18080/acid/acid_db_setup.php

按照提示，建立 acid 运行必需的数据库，如图 8-25 所示。

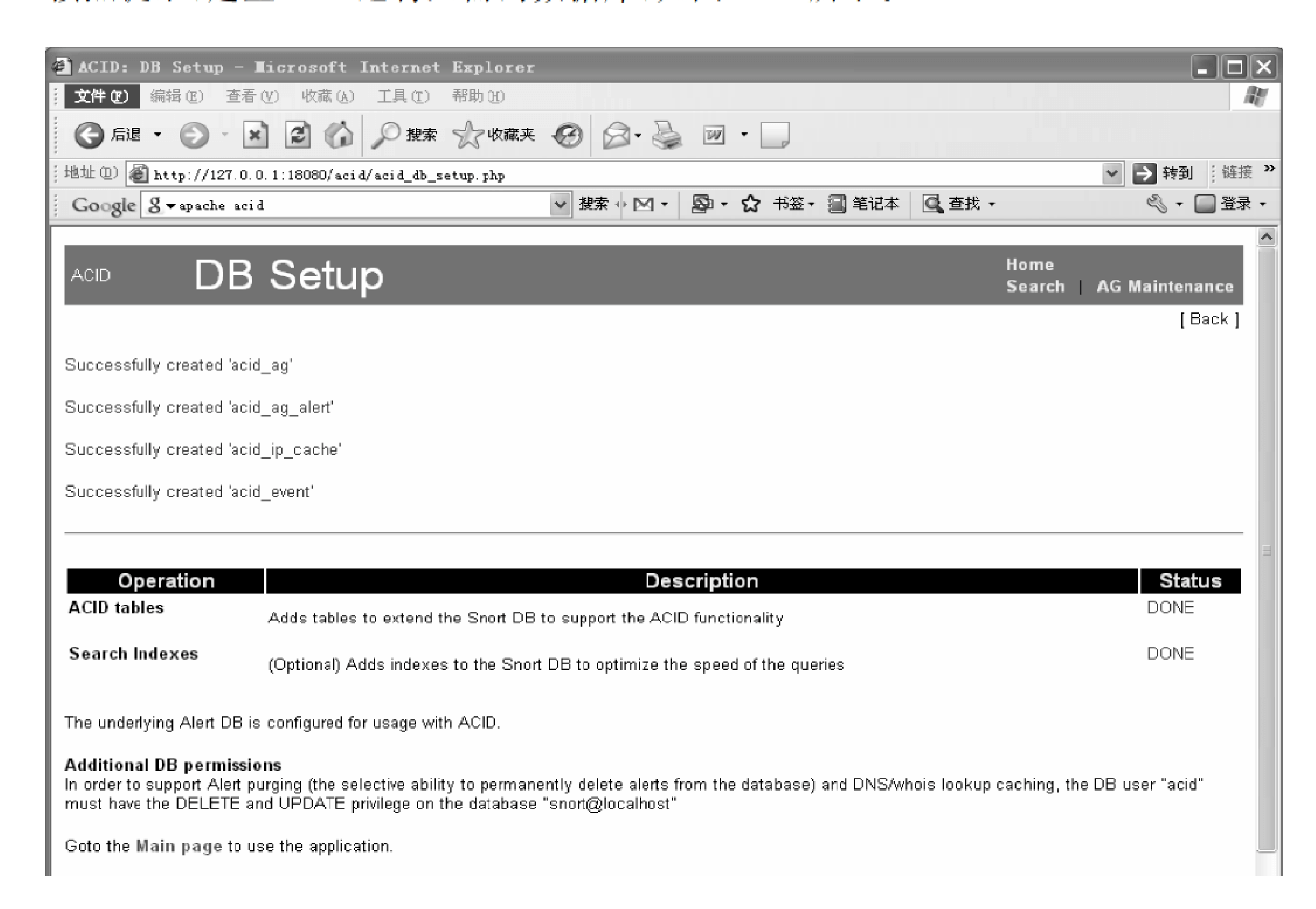


图 8-25　建立 acid 运行必需的数据库

6. 安装 jpgraph-1.26.tar.gz

解压缩 jpgraph-1.26.tar.gz 至 C:\php\jpgraph；修改 C:\php\jpgraph\src 下的文件 jpgraph.php 文件，去掉语句＃DEFINE("CACHE_DIR","/tmp/jpgraph_cache")的注释"；"。

7. 配置 Snort

编辑 C：\snort\etc\snort. conf，进行如下修改。

（1）“var RULE_PATH../rules”修改为 C：/snort/rules(配置 rules 的绝对路径)。

（2）将“include classification. config”改为“include C：\snort\etc\classification. config”(classification. config 的绝对路径)；将“include refrence. config”改为“include C：\snort\etc\reference. config”。

（3）配置 snort 的输出插件：

添加 output database：alert，mysql，host=localhost port=3306 user=root password=123 dbname = snort sensorname = n encoding = hex detail=full；如果 MySQL 与 Snort 不在同一台机器上，则 host 使用 MySQL 所在的主机 IP。

（4）将下面的行号前的“#”去掉，如图 8-26 所示。

```
592 include $RULE_PATH/nntp.rules
593 include $RULE_PATH/other-ids.rules
594 include $RULE_PATH/web-attacks.rules
595  include $RULE_PATH/backdoor.rules
596  include $RULE_PATH/shellcode.rules
597 include $RULE_PATH/policy.rules
598  include $RULE_PATH/porn.rules
599  include $RULE_PATH/info.rules
600  include $RULE_PATH/icmp-info.rules
601  include $RULE_PATH/virus.rules
602  include $RULE_PATH/chat.rules
603  include $RULE_PATH/multimedia.rules
604  include $RULE_PATH/p2p.rules
```

图 8-26　去掉“#”

命令行窗口运行：C：\snort\bin>snort-i3-c "c：\snort\etc\snort. conf"-l "c：\snort\log" -d-e-v，如图 8-27 所示可知配置成功。

```
C:\Snort\bin>snort -i3 -c "c:\snort\etc\snort.conf" -l "c:\snort\log" -d -e -X
Running in IDS mode
Log directory = c:\snort\log

Initializing Network Interface \Device\NPF_{26B41D7B-1C18-41F0-854E-9E7A6D15D817}

        --== Initializing Snort ==--
Initializing Output Plugins!
Decoding Ethernet on interface \Device\NPF_{26B41D7B-1C18-41F0-854E-9E7A6D15D817}
Initializing Preprocessors!
Initializing Plug-ins!
Parsing Rules file c:\snort\etc\snort.conf

++++++++++++++++++++++++++++++++++++++++++++++++++++
Initializing rule chains...
```

图 8-27　配置成功

其中：-X 参数用于在数据链路层记录 raw packet 数据；

-i3 参数表示监测主机的 ID 为 3 的网络接口；

-d 参数记录应用层的数据；

-e 参数显示/记录第二层报文头数据；

-c 参数用于指定 Snort 的配置文件的路径。

在浏览器中输入 http：//127. 0. 0. 1：18080/acid/acid_main. php，进入 acid 分析控制台主界面。如果整个过程配置成功，则出现如图 8-28 所示的界面。

8.5.3　Snort For Windows 的工作模式

在使用 Snort 之前，首先要查看主机上的物理网卡，可以使用命令 Snort -W，如图 8-29 所示，可知这台机器有三个网卡，因为在主机上装有 VMWare 虚拟机，且其中 1、2 为虚拟机的网卡，ID 为 3 的网卡是物理网卡。使用 snort -i3，可以监听第 3 块网卡。

Snort 有三种工作模式，分别为嗅探器模式、数据包记录器模式、网络 IDS 模式，主要用

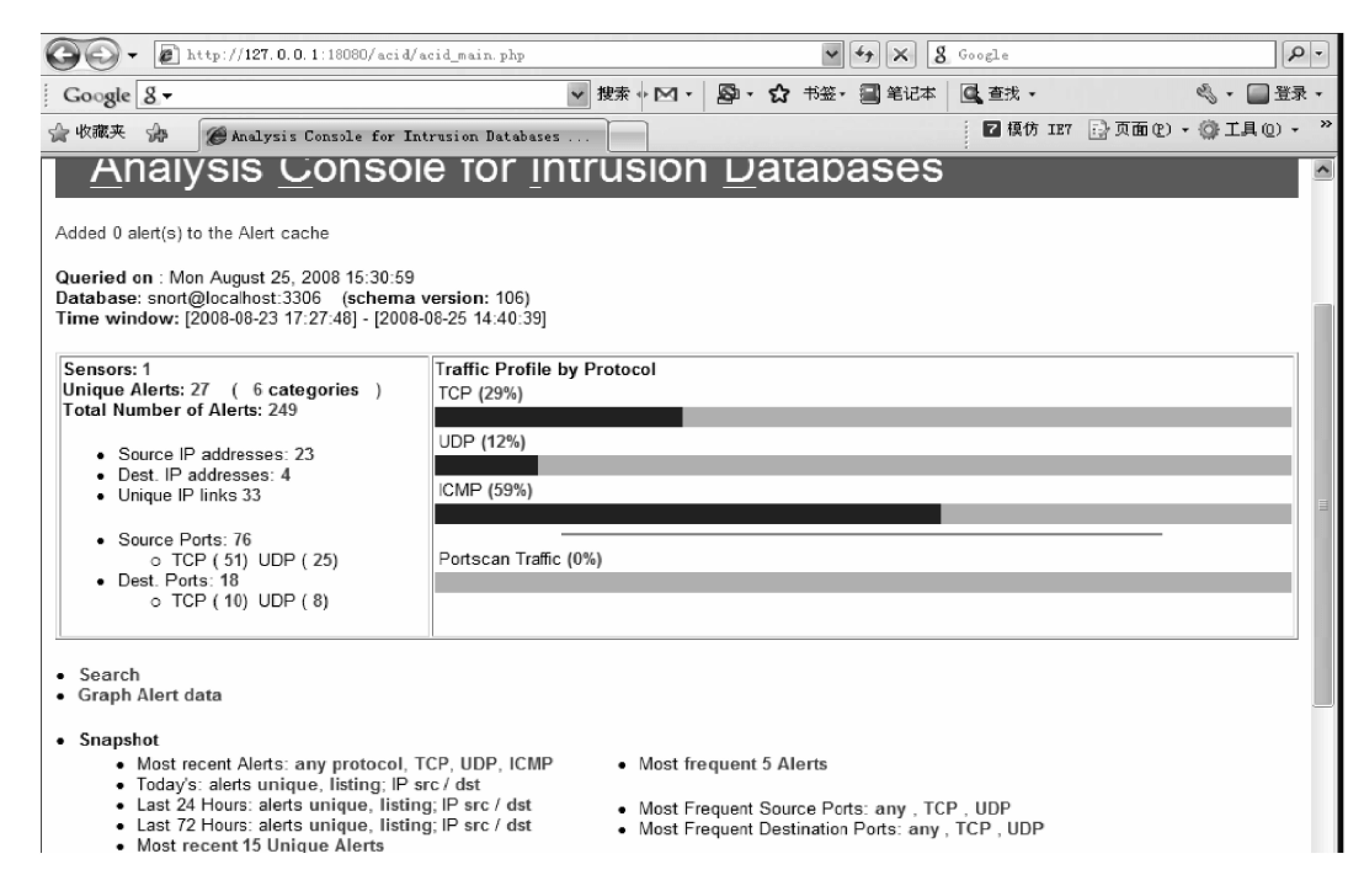


图 8-28　输入 http://127.0.0.1:18080/acid/acid_main.php



图 8-29　查看主机上的物理网卡

Snort 作为 IDS,但同时也要对其他两种模式有所了解。

（1）Snort 的嗅探模式：通常可以用来检测 Snort 是否安装成功，输入命令 Snort -v -i3，使得 Snort 只将 IP 和 TCP/UDP/ICMP 的包头信息输出到屏幕上，如图 8-30 所示。

```
08/25-15:50:09.427533 202.119.201.73 -> 211.136.199.76
ICMP TTL:128 TOS:0x0 ID:4817 IpLen:20 DgmLen:74
Type:3  Code:3  DESTINATION UNREACHABLE: PORT UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
211.136.199.76:42163 -> 202.119.201.73:3665
UDP TTL:109 TOS:0x0 ID:48696 IpLen:20 DgmLen:46
Len: 18
** END OF DUMP
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=

08/25-15:50:11.117073 219.133.49.7:80 -> 202.119.201.73:1026
TCP TTL:88 TOS:0x0 ID:19514 IpLen:20 DgmLen:106
***AP*** Seq: 0x684D2774  Ack: 0x1710258C  Win: 0x578  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=

08/25-15:50:11.155916 219.133.49.7:80 -> 202.119.201.73:1026
TCP TTL:88 TOS:0x0 ID:19515 IpLen:20 DgmLen:106
***AP*** Seq: 0x684D27B6  Ack: 0x1710258C  Win: 0x578  TcpLen: 20
```

图 8-30　Snort 的嗅探模式

（2）数据包记录器模式：如果要将数据包记录在 LOG 文件中（C：\snort\log），则使用下面的命令：

```
Snort -d -e -v -i3 -l c:\snort\log
```

其中：-l 选项指定了存放日志的文件夹；运行该命令后，将在 C：\snort\log 目录下自动生成许多文件夹和文件，文件夹以数据包目的主机的 IP 地址命名，如图 8-31 所示，每个文件夹记录的是和该外部主机相关的网路流量。

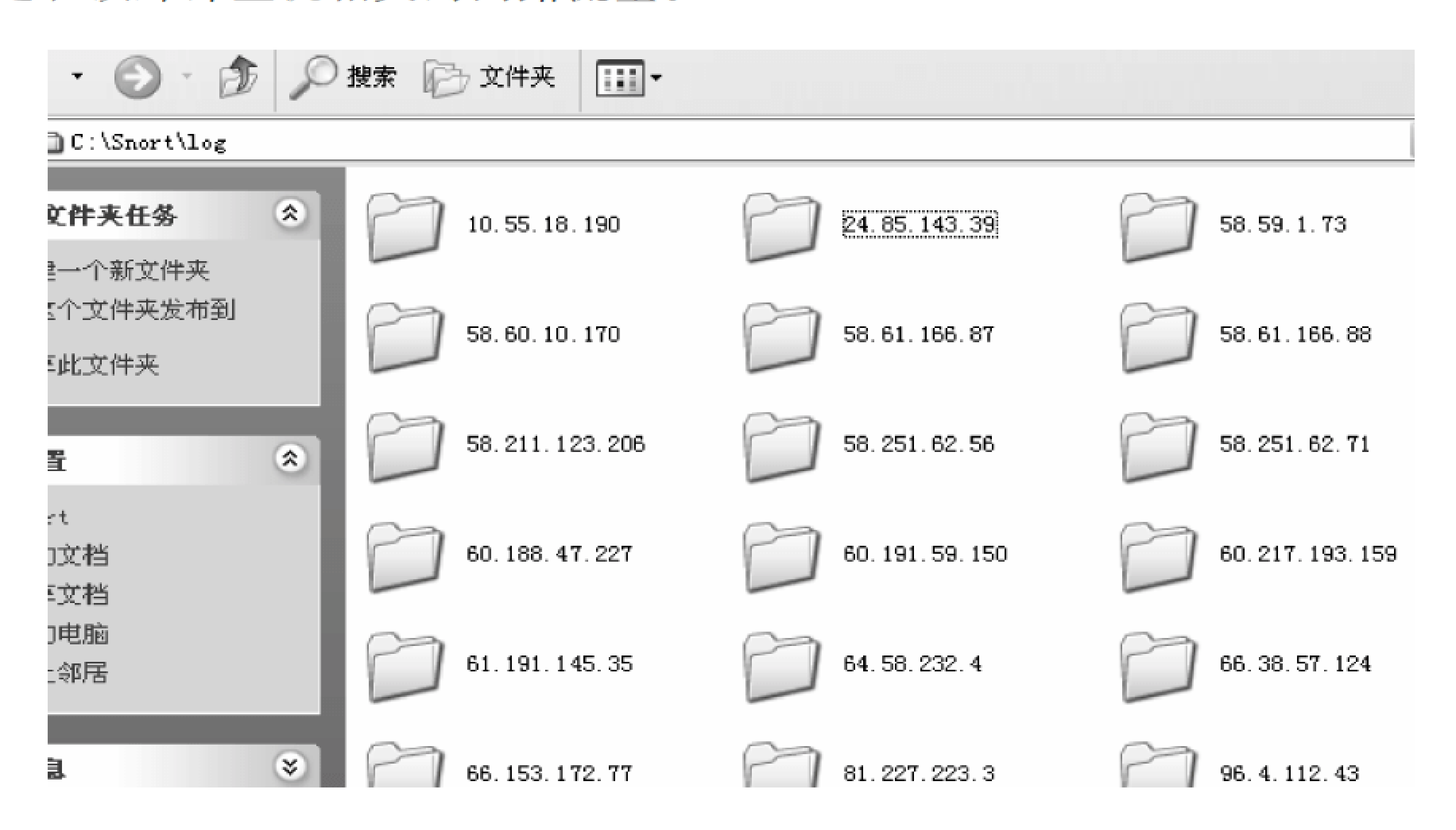


图 8-31　数据包记录器模式

打开日志文件，其记录的格式与嗅探器下的屏幕输出类似，为 Tcpdump 格式，如图 8-32 所示。

```
08/25-16:11:46.940722 0:1C:25:CA:62:E8 -> 0:D0:F8:BC:E7:B type:0x800 len
202.119.201.73:6002 -> 58.61.166.87:8000 UDP TTL:128 TOS:0x0 ID:2343 IpL
Len: 64
03 04 2C 62 00 00 00 01 00 00 00 00 00 00 00 00  ..,b............
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 01 00 00 12 21 00 00 00  ............!...
00 00 00 00 00 00 00 08 01 00 00 01 09 C8 A3 B5  ................

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

08/25-16:11:46.972471 0:D0:F8:BC:E7:B -> 0:1C:25:CA:62:E8 type:0x800 len
58.61.166.87:8000 -> 202.119.201.73:6002 UDP TTL:50 TOS:0x0 ID:0 IpLen:2
Len: 56
03 04 DD 90 00 00 00 01 00 00 00 00 00 00 00 02  ................
```

图 8-32　打开日志文件

（3）IDS 模式：使用命令 snort -d -e -v -i3 -l c：\snort\log-c c：\snort\etc\snort.conf 可以启动 IDS 模式，-c 选项表示使用 snort.conf 中的规则集。Snort 会将每个包和规则集进行匹配，如符合规则就采取规则所指定的动作。在下一部分，将使用 Snort 的 IDS 模式检测攻击。

8.5.4　使用搭建的 Windows 入侵检测平台检测入侵

我们在主机 202.119.201.73 上配置了入侵检测平台，使用 MySQL 来存储检测的信息，使用 acid 作为检测控制台的主界面。

启动 Snort，使用命令 C：\Snort\bin>snort -dve -l c：\snort\log -h 202.119.201.65/24 -c c：\snort\etc\snort.conf。

使用 X-Scan v3.3 在主机 202.119.201.78 上对网段 202.119.201.65～202.119.201.100 进行扫描，如图 8-33 所示。

X-Scan v3.3 GUI

文件(V) 设置(W) 查看(X) 工具(Y) Language 帮助(Z)

202.119.201.73 (Microso
202.119.201.75 (IBM AIX
202.119.201.77
202.119.201.78 (Windows
202.119.201.95 (IBM AIX

主机	累计时间	插件时间	活动线程	当前进度
202.119.201.73	136	75	9	正在执行漏洞检测脚本"sapdb_detect.nasl" ... (778/3
202.119.201.75	136	28	0	正在检测"漏洞检测脚本" ...
202.119.201.77	136	110	1	正在检测"NT-Server弱口令: Administrator/!@#$%^&" ..
202.119.201.78	136	88	5	正在执行漏洞检测脚本"smb_group_replicator.nasl" ...
202.119.201.95	115	16	0	正在检测"漏洞检测脚本" ...

普通信息 | 漏洞信息 | 错误信息

[202.119.201.78]: "IIS编码/解码漏洞"扫描完成.

图 8-33　扫描网段 202.119.201.65～202.119.201.100

打开 acid 检测控制台主界面 http：//127.0.0.1：18080/acid/acid_main.php，如图 8-34 所示，分别给出了 TCP、UDP、ICMP 协议的详细日志情况，点击后面的百分比可以查看详细信息。

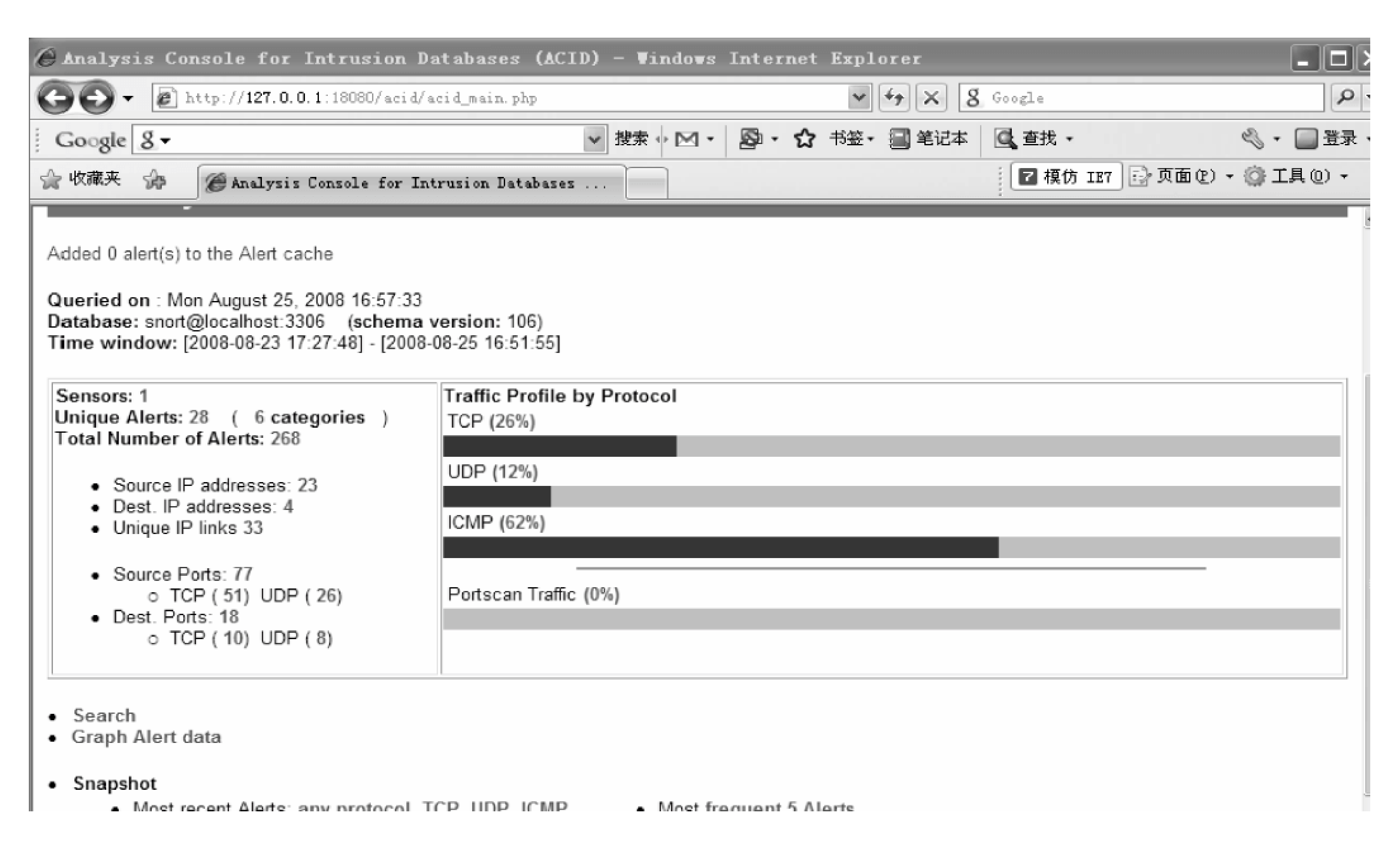
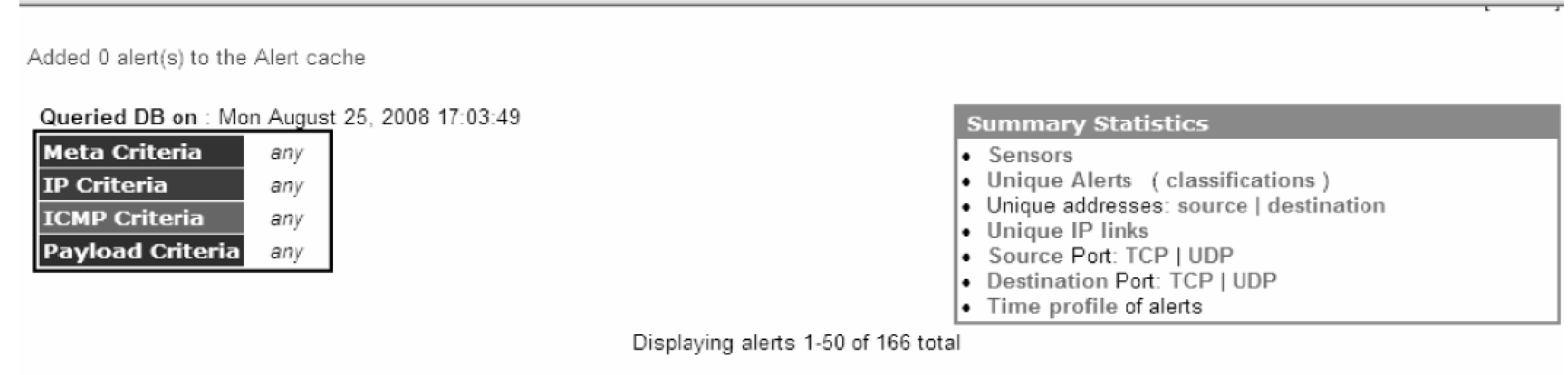


图 8-34　检测控制台主界面

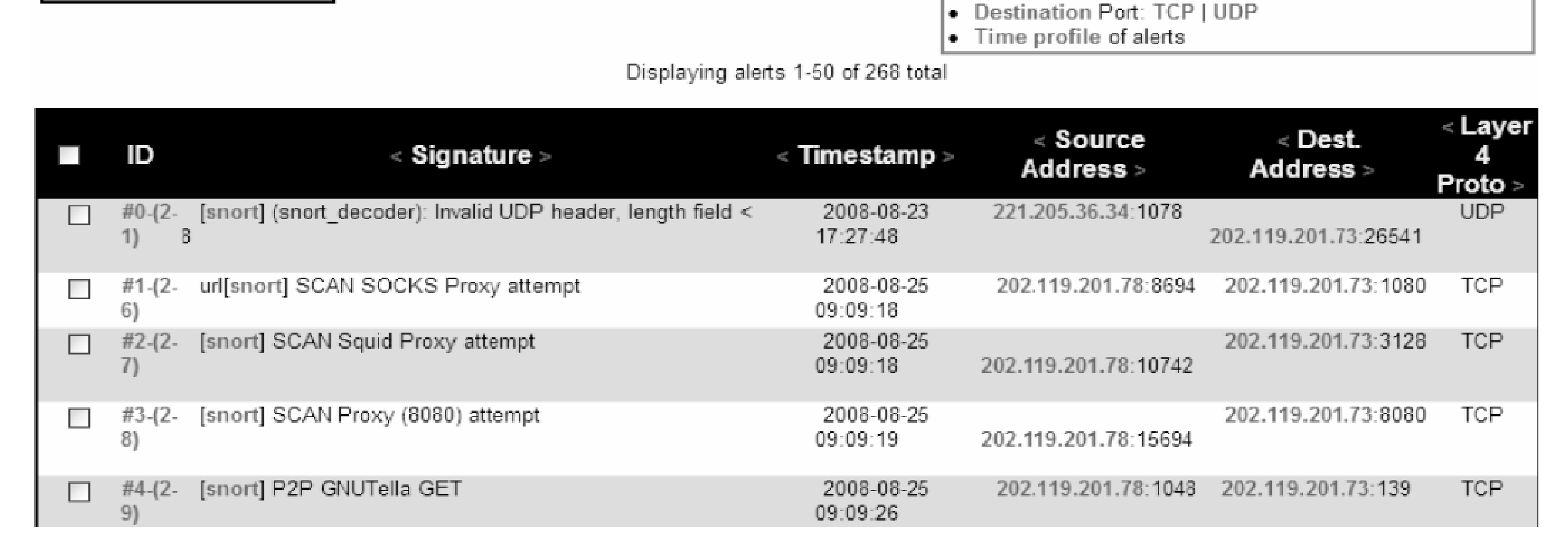
ICMP 协议日志给出了详细的信息，如时间戳、目的/源地址，如图 8-35 所示。

单击 alert 后面的数字，则可详细地看到所给出的特殊的流量记录，这些可能就是某种攻击，如图 8-36 所示。



	ID	Signature	Timestamp	Source Address	Dest. Address	Layer 4 Proto
☐	#0-(2-268)	[snort] ICMP Destination Unreachable (Port Unreachable)	2008-08-25 16:51:55	202.119.201.73	202.119.201.78	ICMP
☐	#1-(2-266)	[snort] ICMP Destination Unreachable (Port Unreachable)	2008-08-25 16:51:45	202.119.201.73	202.119.201.78	ICMP
☐	#2-(2-267)	[snort] ICMP Destination Unreachable (Port Unreachable)	2008-08-25 16:51:45	202.119.201.73	202.119.201.78	ICMP
☐	#3-(2-265)	[snort] ICMP Destination Unreachable (Port Unreachable)	2008-08-25 16:51:43	202.119.201.73	202.119.201.78	ICMP
☐	#4-(2-263)	[arachNIDS][snort] ICMP PING Windows	2008-08-25 16:51:40	202.119.201.78	202.119.201.73	ICMP
☐	#5-(2-264)	[snort] ICMP Echo Reply	2008-08-25 16:51:40	202.119.201.73	202.119.201.78	ICMP
☐	#6-(2-261)	[arachNIDS][snort] ICMP PING Windows	2008-08-25 16:51:39	202.119.201.78	202.119.201.73	ICMP
☐	#7-(2-262)	[snort] ICMP Echo Reply	2008-08-25 16:51:39	202.119.201.73	202.119.201.78	ICMP

图 8-35　ICMP 协议日志



	ID	Signature	Timestamp	Source Address	Dest. Address	Layer 4 Proto
☐	#0-(2-1)	[snort] (snort_decoder): Invalid UDP header, length field < 8	2008-08-23 17:27:48	221.205.36.34:1078	202.119.201.73:26541	UDP
☐	#1-(2-6)	url[snort] SCAN SOCKS Proxy attempt	2008-08-25 09:09:18	202.119.201.78:8694	202.119.201.73:1080	TCP
☐	#2-(2-7)	[snort] SCAN Squid Proxy attempt	2008-08-25 09:09:18	202.119.201.78:10742	202.119.201.73:3128	TCP
☐	#3-(2-8)	[snort] SCAN Proxy (8080) attempt	2008-08-25 09:09:19	202.119.201.78:15694	202.119.201.73:8080	TCP
☐	#4-(2-9)	[snort] P2P GNUTella GET	2008-08-25 09:09:26	202.119.201.78:1048	202.119.201.73:139	TCP

图 8-36　特殊的流量记录

8.6　其他类型的入侵检测系统

8.6.1　文件完整性检查系统

文件完整性检查系统检查计算机中上次检查后文件变化的情况。文件完整性检查系统保存有每个文件的数字摘要数据库,每次检查时,它重新计算文件的数字摘要并将它与数据库中的值相比较,若不同,则说明文件已经被修改;若相同,则说明文件未发生变化。

文件的数字摘要通过 Hash 函数计算得到。不管文件长度如何,它的 Hash 函数计算结果是一个固定长度的数字。与加密算法不同,Hash 算法是一个不可逆的单向函数。采用

安全性高的 Hash 算法(如 MD5、SHA)时,两个不同的文件几乎不可能得到相同的 Hash 结果。从而,一旦文件被修改,就可以检测出来。

文件完整性检查系统有以下优点。

(1) 从数学上分析,攻克文件完整性检查系统,无论是时间上还是空间上都是不可能的。文件完整性检查系统是非常强有力的检测文件被修改的工具。事实上,文件完整性检查系统是检测系统被非法使用的最重要的工具之一。

(2) 文件完整性检查系统具有相当的灵活性,可以配置为检测系统中的所有文件或某些重要文件。

(3) 当一个入侵者攻击系统时,首先,他要掩盖他的踪迹,即通过更改系统中的可执行文件、库文件或日志文件来隐藏他的活动;其次,他要做一些改动保证下次能够继续入侵。这两种活动都能够被文件完整性检查系统检测到。

尽管如此,文件完整性检查系统也存在一些不足。

(1) 文件完整性检查系统依赖于本地的摘要数据库。与日志文件一样,这些数据可能被入侵者修改。当一个入侵者取得管理员权限后,在完成破坏活动后,可以运行文件完整性检查系统更新数据库,从而骗过系统管理员。当然,也可以将完整数据库放在只读的介质上,但这样的配置不够灵活性。

(2) 做一次完整的文件完整性检查是一件非常耗时的工作,在 Tripwire 中,在需要时可选择检查某些系统特性而不是完全的摘要,从而加快检查速度。

(3) 系统有些正常的更新操作(如软件升级)可能会带来大量的文件更新,从而产生比较繁杂的检查与分析工作。

8.6.2 日志文件监视

日志文件通常有应用程序日志,安全日志、系统日志、DNS 服务器日志、FTP 日志、WWW 日志等,下面以 FTP 为例说明日志记录的结构。

FTP 日志和 WWW 日志默认情况,每天生成一个日志文件,包含了该日的一切记录,文件名通常为 ex(年份)(月份)(日期),例如 ex001023,就是 2000 年 10 月 23 日产生的日志,用记事本就可直接打开,如下例:

```
#Date: 20001023 0315(服务启动时间日期)
#Fields:time cip csmethod csuristem scstatus
0315 127.0.0.1[1]USER administator 331(IP 地址为 127.0.0.1 用户名为 administator 试图登录)
0318 127.0.0.1[1]PASS - 530(登录失败)
032: 04 127.0.0.1[1]USER nt 331(IP 地址为 127.0.0.1 用户名为 nt 的用户试图登录)
032: 06 127.0.0.1[1]PASS - 530(登录失败)
032: 09 127.0.0.1[1]USER cyz 331(IP 地址为 127.0.0.1 用户名为 cyz 的用户试图登录)
0322 127.0.0.1[1]PASS - 530(登录失败)
0322 127.0.0.1[1]USER administrator 331(IP 地址为 127.0.0.1 用户名为 administrator 试图登录)
0324 127.0.0.1[1]PASS - 230(登录成功)
0321 127.0.0.1[1]MKD nt 550(新建目录失败)
0325 127.0.0.1[1]QUIT - 550(退出 FTP 程序)
```

从日志里就能看出 IP 地址为 127.0.0.1 的用户一直试图登录系统,换了四次用户名和密码才成功,管理员立即就可以得知管理员的入侵时间、IP 地址以及探测的用户名,以此为

依据，考虑更换此用户名的密码，或者重命名 administrator 用户。

Microsoft Windows 中的事件记录提供了一种标准的、集中化的方式使得应用程序记录重要的软件和硬件事件。“事件”，按 Windows 中的定义，是指发生的任何需要通知用户的重要事情(无论是在操作系统中还是在应用程序中)，每个事件日志条目都可以分为以下几个类别之一：错误、警告、信息、成功审核和失败审核(详情见表 8-1)。

表 8-1　Windows 日志条目

条目类型	内　容
信息(Information)	信息事件指示不常发生但很重要的操作。例如，当 Microsoft SQL Server 成功加载时，记录一个“已启动 SQL Server”。信息事件应当是恰当的。注意，这对于主要服务器服务是正确的行为，但对于桌面程序来说，每次启动时都记录一个事件一般来说是不恰当的
警告(Warning)	警告事件指示不会马上就很明显的问题，但此问题以后可能会带来麻烦。资源消耗是警告事件的一个很好的例子。例如，如果磁盘空间不足，应用程序会记录一个警告事件。如果一个应用程序可以从事件中恢复而未丢失功能或数据，它一般会将此事件归类为警告事件
错误(Error)	错误事件指示用户应知道的重大问题，如功能或数据的丢失。例如，如果某个服务在系统引导时不能加载，它就会记录一个错误事件
成功审核(SuccessAudit)	成功审核事件是一个安全性事件，在经审核的访问尝试成功时发生。例如，成功的登录尝试就是一个成功审核事件
失败审核(FailureAudit)	失败审核事件是一个安全性事件，在经审核的访问尝试失败时发生。例如，尝试打开文件失败就是一个失败审核事件

目前广泛使用的 Windows 操作系统主要是 Windows 2000，Windows XP，Windows 2003 Server，它们都是基于 Windows NT 内核开发的。Windows NT 的日志文件一般分为三类。

(1) 系统日志(System)：跟踪各种各样的系统事件，记录由 Windows NT 系统组件产生的事件。例如，在启动过程加载驱动程序错误或其他系统组件的失败记录在系统日志中。

(2) 应用程序日志(Application)：记录由应用程序或系统程序产生的事件，比如应用程序产生的装载 DLL(动态链接库)失败的信息将出现在日志中。

(3) 安全性日志(Security)：记录登录上网、下网、改变访问权限以及系统启动和关闭等事件以及与创建、打开或删除文件等资源使用相关联的事件。

这三个日志文件通常保存在%Systemroot%\System32\config 文件夹下，名称分别为 SysEvent.evt，AppEvent.evt 和 SecEvent.evt。当然这个位置并不是固定的，系统管理员可以通过修改注册表来改变它们的保存路径，根据指向具体文件的配置，可以在 HKLM_LOCAL_MACHINE\System\CurrentControlSet\Services\Eventlog 目录下看到三个日志。

Windows 使用了一种特殊的格式保存它们并加以保护，一般情况下普通用户不能随意更改。不能用针对普通 TXT 文件的编辑方法来编辑它们，Word 类的软件也奈何不了。甚至不能对它们进行“重命名”、“删除”或“移动”操作，否则系统拒绝对文件的访问。对事件日志的访问权限是由应用程序运行所使用的账户来决定的。“本地系统”账户是服务应用程序

可以使用的专用账户。管理员账户由系统的管理员组成。“服务器操作员”账户(ServerOp)由域服务器的管理员组成。“全体”账户包括所有系统上的所有用户。表 8-2 说明了对各日志具有读取、写入和清除访问权限的账户。

表 8-2 日志权限管理

日　志	账　户	访问权限
应用程序	本地系统	读取，写入，清除
	管理员	读取，写入，清除
	服务器操作员	读取，写入，清除
	全体	读取，写入
安全性	本地系统	读取，写入，清除
	管理员	读取，写入
	全体	无
系统	本地系统	读取，写入，清除
	管理员	读取，写入，清除
	全体	读取，清除
	服务器操作员	读取

另外，如果用户已获准具有下列权限之一，他们就可以读取和清除“安全性”日志：“管理审核和安全日志”用户权限；SE_AUDIT_NAME 特权。

8.7 蜜　罐

8.7.1 蜜罐的概念

“蜜罐”这一概念最初出现在 1990 年出版的一本小说 *The Cuckoo's Egg* 中，在这本小说中描述了作者作为一个公司的网络管理员，如何追踪并发现一起商业间谍案的故事。“蜜网项目组”(The Honeynet Project)的创始人 Lance Spitzner 给出了对蜜罐的权威定义：蜜罐是一种安全资源，其价值在于被扫描、攻击和攻陷。这个定义表明蜜罐并无其他实际作用，因此所有流入/流出蜜罐的网络流量都可能预示了扫描、攻击和攻陷。而蜜罐的核心价值就在于对这些攻击活动进行监视、检测和分析。蜜罐并非一种安全解决方案，这是因为蜜罐并不会“更正”任何错误。

蜜罐是一种网络攻击诱骗工具，又叫网络陷阱，它通过模拟一个或多个易受攻击的目标系统，给黑客提供一个包含漏洞并容易被攻破的系统作为他们的攻击目标。其主要作用是吸引入侵者来嗅探、攻击，同时不被察觉地将他们的活动记录下来，进而评估黑客攻击的目的、使用的工具、运用的手段、造成的后果。通常蜜罐部署在容易被入侵者发现的地方，辅以各种网络攻击诱骗技术来诱使入侵者上当，让入侵者误以为已经成功侵入网络系统，而系统则完成了对入侵者的攻击过程的记录，通过这些记录可以分析、学习入侵者侵入系统所使用的工具，采用的手段、技术和他们的意图。有些蜜罐系统甚至可以对攻击者网上聊天的内容进行记录。蜜罐系统管理人员通过分析攻击者的网上聊天内容还可以获得攻击者的活动范

围以及下一个攻击目标，根据这些信息，管理人员可以提前对系统进行保护。同时某种程度上，在蜜罐系统中记录的信息还可以作为对攻击者进行起诉的证据，这也是蜜罐设计的初衷之一。同时蜜罐可以吸引和转移攻击者的注意，延缓他们对真正目标的攻击，消耗他们的时间和资源，影响他们的意志。甚至可以事先掌握攻击者的行为，跟踪攻击者，并有效地制止攻击者的破坏行为，形成威慑攻击者的力量。蜜罐还可以对攻击进行检测和报警，和入侵检测系统相比，由于它不提供正常的网络服务，任何对蜜罐的网络访问都被认为是攻击，所以蜜罐的一大优点就是不会产生误报和漏报。

8.7.2　蜜罐的原理

从蜜罐的定义可以看出，蜜罐的目的就是布置安全陷阱，诱骗攻击者对它攻击，在攻击者认为它成功达到攻击目的的同时，蜜罐系统对整个攻击过程也进行了记录。下面就从布置安全陷阱、诱骗攻击和记录攻击过程三个方面来叙述蜜罐的原理。

1. 布置安全陷阱

如何巧妙地布置安全陷阱是蜜罐系统非常关键的一个步骤，如果攻击者很容易识别陷阱或根本不知道陷阱位于何处，蜜罐系统的作用也就无用武之地了。在一个典型的受保护网络中，蜜罐系统通常位于受保护网络内，它不对外提供 IP 地址，也不对外提供任何网络服务及服务接口，因此在这样的系统中，任何对该蜜罐主机的连接及操作都可以认为是攻击的发生。在布置安全陷阱方面，可以与生活中陷阱的概念相对比，猎人的陷阱通常都会放置于一个猎物经常出没的地方或在陷阱旁设置一些新鲜的事物来引起猎物好奇的天性，这样往往可以成功捕获猎物。

2. 诱骗攻击

当攻击者成功入侵受保护网络后，如何诱骗攻击者只对蜜罐攻击而暂时忽略对其他安全主机的攻击呢？这正是诱骗攻击的目的，也是蜜罐系统与受保护主机在安全设置上的区别。在一个比较典型的攻击过程中，当攻击者攻入一个受保护网络后，它首先会寻找一台安全措施比较弱，易于进行攻击的系统进行突破，因为攻击者一般会认为只要将受保护网络的一台机器成功攻破，再基于此攻破主机对内网的其他主机进行攻击将会更加容易。其实这样做的道理也是明显的，因为内网的相互访问可以绕过众多的安全检查，一般认为是安全的。因此一旦攻击者找到了最易攻破的入口点，对其他主机的攻击也就易如反掌。基于这样的原理，只要蜜罐系统的诱骗技术与真实主机几乎无异，则大多时候可以成功地将攻击者引入蜜罐陷阱。

3. 记录攻击过程

攻击过程的记录存在于实施攻击的整个过程，它通过对攻击过程的详细跟踪来实现对攻击过程的特征提取。同样的，有的攻击者如果在攻击过程中发现所攻击系统是蜜罐系统，可能会寻找蜜罐系统的弱点而将蜜罐系统的日志删除，因此在日志记录过程中，要谨防蜜罐系统中日志记录模块的安全性。

8.7.3　蜜罐分类

对蜜罐的分类方式有以下三种。

（1）蜜罐从性质上可以分为产品型蜜罐和研究型蜜罐。

产品型蜜罐的目的在于为一个组织的网络提供安全保护，包括检测攻击、防止攻击造成破坏及帮助管理员对攻击做出及时正确的响应等功能。较具代表性的产品型蜜罐包括DTK、honeyd等开源工具和KFSensor、ManTraq等一系列的商业产品。产品型蜜罐一般用来作为诱饵把黑客的攻击尽可能长时间地捆绑在蜜罐上，赚得时间保护实际网络环境，有时也用来收集证据作为起诉黑客的依据。

研究蜜罐不保障某个系统或团体的安全，也不为某个系统或团体服务，而是旨在收集网络中各种安全威胁的信息，为研究这些安全威胁服务。研究蜜罐主要通过捕捉网络入侵者的活动来实现其目的。

蜜罐可以是单独的系统，也可以和受保护的关键系统放在一起。研究型蜜罐通常是一个单独的系统，部署在网络中易于被网络黑客发现的地方；产品型蜜罐通常和受保护的关键系统放置在一起，作为传统信息安全保障手段的辅助手段。首先，蜜罐尤其是研究型蜜罐需要使用各种攻击诱骗手段吸引黑客的注意并诱使其对蜜罐发起攻击。然后，蜜罐需要构造一个或多个存在安全漏洞的目标系统用于与入侵者进行直接的交互和斡旋，通过多层数据捕捉、收集手段和技术对入侵者的活动进行监视和记录，并通过网络备份到远程日志服务器；同时运用多种控制手段和技术对入侵者的活动进行限制，防止蜜罐为入侵者利用作为攻击其他系统的跳板。最后利用各种工具和手段对入侵者的活动进行分析，对入侵者的技术、手段、意图进行研究和学习。产品型蜜罐的主要目的是消耗入侵者的时间，延缓入侵者对关键系统的攻击，将入侵者对关键系统的破坏减小到最低。

（2）根据蜜罐系统与攻击交互的频繁程度可将蜜罐系统分为低交互蜜罐、中交互蜜罐和高交互蜜罐。

低交互蜜罐只为外界攻击提供非常有限的应答或只是简单的监听网络的连接信息，如可能只开设几个虚假的端口信息，当攻击者根据扫描工具找到这些端口并进行攻击时，蜜罐系统因为没有回应，因而只能产生一些简单的日志信息，而这些信息只是一些初始的连接信息，无法真正评估攻击的目的，而且因为无法回应或只回应少量信息，聪明的黑客很快就会发现异常并停止攻击，这样也就失去了诱骗的初衷。当然，因为交互信息的有限性，这种类型的蜜罐系统的安全性也是最高的，它不易被黑客攻破而成为攻击其他内网主机的跳板。

中交互蜜罐为外界提供近乎真实的服务应答机制，但因为交互频率的提高，蜜罐系统可以提供较低交互系统更加详细的日志和攻击手段。但中交互系统因为要模拟众多的服务而加大了其普通使用的技术门槛。因为在无操作系统支持的基础上提供各种协议服务，而且要诱骗攻击者认为这是一个真正的服务，这就要求在技术上要非常的逼真与高效，攻击者往往是对网络安全技术有深入了解的人，如果只是简单的或偶尔不正确的响应一些攻击者的请求，攻击者可能很快就会发现被攻击系统是个陷阱，从而停止攻击。因此架设一个中交互系统是比较复杂的，要求实施者对网络协议与服务有深刻的认识。同时，因为中交互系统提供了较多的网络服务，有可能被攻击者利用成为跳板，因此，在实施该系统的网络内，应经常检查系统日志，并严格检查是否有安全漏洞已经被黑客利用。

高交互蜜罐是为外界提供真实的操作系统服务的蜜罐系统。因为在现实的网络攻击中，最可怕的并不是攻击者的入侵与破坏，而是当攻击者攻破系统并为所欲为时，管理员仍

然无法知道攻击者是如何闯入安全系统中的。因此，在这样的要求下，高交互系统显示出了其独特的优点，因为高交互系统可以提供真实的操作系统环境，攻击者完全可以认为它就是一台真实的服务器主机，因此，它可以对攻击者的攻击方法进行详细的日志记录。当攻击者自认为成功入侵时，系统也成功的记录了它的入侵过程，这样可以帮助系统发现当前系统漏洞并采取相应措施修补漏洞。当然，因为部署了真实的操作系统，就难免有攻击者可以通过蜜罐系统对其他内网主机进行攻击，因此，管理员应定期检查蜜罐主机的安全性和完整性，也可通过其他方法将蜜罐主机与受保护主机隔离，使攻击者即使攻破了蜜罐系统也无法跳板攻击其他主机。

（3）从具体实现的角度，可以分为物理蜜罐和虚拟蜜罐。

高交互蜜罐通常是一台或多台拥有独立 IP 和真实操作系统的物理机器，提供部分或完全真实的网络服务，这种蜜罐叫物理蜜罐。中低交互的蜜罐可以是虚拟的机器、虚拟的操作系统、虚拟的服务，这样的蜜罐，就是虚拟蜜罐。配置高交互性的物理蜜罐成本很高，相对而言虚拟蜜罐需要较少的计算机资源和维护费用。

8.7.4　蜜罐的优缺点

蜜罐系统有其他网络安全技术没有的几个独特的优势。

第一个优势是蜜罐的数据价值高。蜜罐收集的数据量很少，但是含金量高，蜜罐的概念决定了可以将噪音降到最低（任何对它的访问都是非法、可疑的），不同于其他的安全工具每天收集的数据达到几 GB，大多数蜜罐每天收集的数据都是几 MB。而这些数据记录的都是扫描、探测、攻击，价值量非常高。

第二个优势是蜜罐的资源消耗少。因为蜜罐需要捕获和监视的网络行为很少，所以一般情况下是不会出现资源耗尽的情况。蜜罐仅仅只需监视对它的连接，所以不会存在网络流量大的压力。蜜罐对资源需求的有限性使得不必花费太多的钱去投资。蜜罐相对于防火墙、IDS 这些安全工具来说，它不需要最新的技术——高容量的 RAM、高速 CPU。这就意味着蜜罐配置不需要消耗太多资源。

第三个优势是蜜罐的设计和配置简单。简单性是蜜罐的最大优势，不需要开发复杂新奇的算法，不需要维护特征数据库，不需要配置规则库。只要配置好蜜罐，把它放在网络中，坐着等待就可以了。对于一个系统而言，越简单，其工作的可靠性越好。

蜜罐虽然有众多的优点，同时它也有缺点，这些缺点主要体现在以下几个方面。

第一个缺点是蜜罐的视野比较狭窄。它只能看见针对它的行为。如果攻击者闯入网络，攻击了多个系统，但是由于没有直接攻击蜜罐，那么蜜罐将无法发现这次攻击。

第二个缺点是蜜罐运行时容易留下指纹。指纹是蜜罐的另外一个缺点，尤其是许多商业蜜罐。指纹的存在使得攻击者能鉴别出蜜罐的存在，因为蜜罐会有一些专业特征和行为。如果黑客发现了某个组织在他的内网中使用了蜜罐，他就有可能对蜜罐发动 spoof 攻击，蜜罐检测到这种 spoof 攻击之后，不断地向管理员发送错误的报警信息，这样黑客就扰乱了视听，他的目标很有可能是一个真实的攻击。

第三个缺点是蜜罐运行时的风险。蜜罐将风险带入了网络环境中。这个风险就是一旦蜜罐被攻击了，它有可能就被攻击者利用作为攻击、渗透其他系统或组织的跳板。

8.8 入侵防御系统

8.8.1 入侵防御系统的概念

入侵防御系统(Intrusion Prevention System,IPS)是近年来新兴的一种网络安全产品。它是由入侵检测系统发展而来,兼有防火墙的一部分功能。IPS 系统包含两大功能模块:防火墙和入侵检测。从功能上讲,IPS 是传统防火墙和入侵检测系统的组合,它对入侵检测模块的检测结果进行动态响应,将检测出的攻击行为在位于网络出入口的防火墙模块上进行阻断。然而,IPS 并不是防火墙和入侵检测系统的简单组合,它是一种有取舍地吸取了防火墙和入侵检测系统功能的一个新产品,其目的是为网络提供深层次的、有效的安全防护。

当前最常见的入侵防御方法是通过网络入侵检测系统和防火墙的联动来实现的。当入侵检测系统发现攻击企图后,它会通知防火墙将攻击来源的 IP 地址或端口禁止。最初联动的概念是防火墙厂商 Check Point 提出的,Check Point 提出了 OPSEC 开放接口,与其他厂商密切合作,实现了 Check Point 防火墙和身份认证、内容安全、入侵检测等产品的互动。现在市场上的主流网络入侵检测产品,例如 NFR NID、SecoShield 等,几乎都可以和 CheckPoint 防火墙进行互动。

目前防火墙加入侵检测产品联动的方式也有一些不足。首先,使用两个产品防御攻击会使系统很复杂,任何一个产品发生故障都会导致安全防范体系的崩溃,而且两个产品的维护成本也比较高。其次,防火墙和入侵检测产品的互动并没有一个被广泛认可的通用标准,大多数安全厂商各行其是,入侵检测产品厂商只与自己感兴趣的防火墙进行互动,而防火墙厂商提出的开放接口却又互不相同。没有一个通用的标准可以让所有的入侵检测产品和所有的防火墙进行互动。这样,用户在采购安全产品的时候,不得不考虑到不同产品的交互性问题,从而限制了用户选择产品的范围。

8.8.2 基于网络的入侵防御系统

网络入侵防御系统(Network-based Intrusion Prevention System,NIPS)与受保护网段是串联部署的。受保护的网段与其他网络之间交互的数据流都必须通过 NIPS 设备。当通信(指网络流量)通过 NIPS 时,通信将被监视是否存在攻击。各种系统的攻击检测机制是不一样的,但是最精确的系统通常集成了多种技术,可以达到很高的攻击和误用检测水准。极高的精确性和高级别的性能,对有效的 NIPS 是至关重要的,因为攻击的误报将导致合法的通信被阻断,也就是可能出现拒绝服务的情形。高性能是保证合法通信通过 NIPS 时不会延迟。当检测到攻击时,NIPS 丢弃或阻断含有攻击性的数据,进而阻断了攻击。

NIPS 兼有 IDS、防火墙和反病毒等安全组件的特性,有时亦被称为内嵌式(inline)IDS 或网关式(gateway)IDS。尽管下一代防火墙——深层检测(Deep Inspection)防火墙还不能成为主流网络安全设施,但也有同 NIPS 相似的特性。正如一个典型的防火墙,NIPS 至少有两个网卡,一个连接内部网络,另一个连接外部网络。当数据包经过任何一个网卡接口,NIPS 将把它们传递到检测引擎。在这一点上,IPS 的检索引擎同任何 IDS 一样,将确定此

包是否包含威胁网络安全的特征。但是，与 IDS 不同的是，当检测到一个恶意的数据包时，IPS 不但发出警报，还可以丢弃此包，或阻断此会话(session)。当该特殊 TCP 会话流的其他数据包到达 NIPS 时，数据包将被立即丢弃。合法数据包通过另一个接口传递到目的主机。

在技术上，NIPS 吸取了目前 NIDS 所有的成熟技术，包括特征匹配、协议分析和异常检测。特征匹配是最广泛应用的技术，具有准确率高、速度快的特点。基于状态的特征匹配不但检测攻击行为的特征，还要检查当前网络的会话状态，避免受到欺骗攻击。协议分析是一种较新的入侵检测技术，它充分利用网络协议的高度有序性，并结合高速数据包捕捉和协议分析，来快速检测某种攻击特征。协议分析正在逐渐进入成熟应用阶段。协议分析能够理解不同协议的工作原理，以此分析这些协议的数据包，来寻找可疑或不正常的访问行为。协议分析不仅仅基于协议标准(如 RFC)，还可基于协议的具体实现，这是因为很多协议的实现偏离了协议标准。通过协议分析，IPS 能够针对插入(insertion)与规避(evasion)攻击进行检测。异常检测的误报率比较高，NIPS 不将其作为主要技术。

总的来说，NIPS 具有以下优点。

(1) 单个通信(流量)控制点可以保护成千的位于 NIPS 下面的系统。这样，组织、企业就可以很快地改变网络规模和更加灵活地改变网络的体系结构。

(2) NIPS 设备像单个探测器(sensor)，易于部署，可以保护成百上千的系统。部署几个或几十个探测器比在成百上千的系统上安装软件，要省去很多的时间和精力。

(3) 提供一个更宽的视野，可以发现威胁情形，例如扫描、探测、攻击基于非单一系统的设备。

(4) 保护非计算机类的网络设备。并非所有的攻击都是针对受 HIPS 保护的、运行操作系统的计算机。例如，路由器、防火墙、VPN 网关、打印机等，都是很容易受攻击的，它们都需要保护。

(5) 与平台无关。NIPS 可以保护所有设备，不管是什么操作系统或应用程序。

(6) 防止网络拒绝服务攻击、分布式拒绝服务攻击、面向带宽的(bandwidth-oriented)攻击、同步洪水(SYN flood)攻击。攻击的一种形式就是向网络发送大量的洪水般的无关的通信，造成网络对授权的(合法的)用户不可用，或网络性能大幅度降低。工作在网络层，NIPS 可以保护系统免受于这些类型的攻击。

8.8.3　基于主机的入侵防御系统

正如基于主机的入侵监测系统，基于主机的入侵防御系统(Host-based Intrusion Prevention System，HIPS)是直接安装在正受保护的机器上的代理(程序)，检测并阻挡针对本机的威胁和攻击。它与操作系统内核紧密捆绑在一起，监视和窃听 API 或到内核的系统调用，阻挡攻击，并记录日志。同时，它还监视针对某一特殊应用的数据流和环境变化，例如 Web 服务器的文件位置及注册表设置，保护应用程序免受那些目前还没有特征的攻击。进出这个特殊系统的通信和应用程序、操作系统的行为将被监视和检查，判断其是否存在攻击迹象。

HIPS 不仅可以保护操作系统，还可以保护在其上运行的应用程序，例如 Web 服务器。当检测到攻击，HIPS 软件程序要么在网络接口层阻断攻击，要么向应用程序或操作系统发出命令，停止攻击所引起的行为。例如，通过禁止恶意程序的执行，可以防止缓冲区溢出攻击。通过拦截和拒绝 IE 发出的写文件命令，可以阻挡攻击者试图通过 IE 这样的应用程序

安装后门程序。

这种方法带来的缺点就是,与主机的操作系统必须紧紧地集成在一起,一旦操作系统升级,将会带来问题。

HIPS 通过在主机/服务器上安装软件代理程序,防止网络攻击入侵操作系统以及应用程序。基于主机的入侵防护能够保护服务器的安全弱点不被不法分子所利用。基于主机的入侵防护技术可以根据自定义的安全策略以及分析学习机制来阻断对服务器、主机发起的恶意入侵。HIPS 可以阻断缓冲区溢出、改变登录口令、改写动态链接库以及其他试图从操作系统夺取控制权的入侵行为,整体提升主机的安全水平。

在技术上,HIPS 采用独特的服务器保护途径,利用由包过滤、状态包检测和实时入侵检测组成分层防护体系。这种体系能够在提供合理吞吐率的前提下,最大限度地保护服务器的敏感内容,既可以以软件形式嵌入到应用程序对操作系统的调用当中,通过拦截针对操作系统的可疑调用,提供对主机的安全防护;也可以以更改操作系统内核程序的方式,提供比操作系统更加严谨的安全控制机制。

由于 HIPS 工作在受保护的主机/服务器上,它不但能够利用特征和行为规则检测,阻止诸如缓冲区溢出之类的已知攻击,还能够防范未知攻击,防止针对 Web 页面、应用和资源的未授权的任何非法访问。HIPS 与具体的主机/服务器操作系统平台紧密相关,不同的平台需要不同的软件代理程序。

HIPS 具有以下优点。

(1) 软件直接安装在系统上,可以保护系统免受攻击,例如阻断程序写文件,阻止用户特权的升级。

(2) 当移动系统依附在受保护网络的外部时,保护它们免受攻击。蠕虫病毒主要是无线上网的笔记本电脑带入受保护网络。为移动的系统装一个 NIPS 是不实际的。

(3) 保护系统免受本地攻击。可以物理(直接)访问系统的人,可以通过执行软盘、CD 或本地的程序,发动本地攻击。这些攻击通常是为了把用户权限提升到超级用户(root)或管理员(administrator)的权限,便于危及网络中的其他系统。

(4) 提供最后一道防线(last line of defense),使系统免受其他安全工具检测的攻击。目标受害者系统本身是安全人员防止系统受危及的最后一个防御点。

(5) 防止相同网段上的系统、设备受到内部攻击或滥用,NIPS 只能保护在不同网段间移动的数据。在同网段系统间发动的攻击只能被 HIPS 检测到。

(6) 保护系统免受于加密的攻击,受保护系统正是加密数据流的终点。HIPS 等加密的数据在本机解密后,检查数据及其行为,或系统的活动。

(7) HIPS 独立于网络体系结构,允许需要保护的系统位于过时的或不常用的网络体系,如令牌环网(token ring),FDDI 等。

习 题 8

1. 什么是入侵检测系统?
2. 试说明入侵检测系统的基本功能。

3. 基于网络的入侵检测系统与基于主机的入侵检测系统各有什么特点？
4. 什么是异常检测？
5. 简单描述一下 Snort 的工作原理。
6. 什么是蜜罐？简单说明一下它的工作原理。
7. 什么是应急响应组？
8. 应急响应计划分为哪几个阶段？
9. 什么是入侵防御系统？
10. 解释一下入侵防御系统的工作原理。

第 9 章　电子邮件安全

随着因特网的普及，电子邮件以其方便、快捷、便宜、容易存储和管理的特点，成为人们联系沟通的重要手段，而电子邮件的安全问题也越来越得到使用者的重视。但现在对电子邮件的攻击现象越来越多，破坏力也越来越强。传统的电子邮件存在的若干不安全因素如邮件可能在不为通信双方所知的情况下被读取、篡改或截获，发信者的身份可能被人伪造等，使得真正重要的信息仍然不宜通过电子邮件来传递。

9.1　电子邮件安全概述

9.1.1　电子邮件的安全需求

安全电子邮件就是运用各种安全模块来保障邮件在 Internet 传送过程中的安全性，通常安全性体现在：

- 机密性：利用收件人的数字证书完成对电子邮件加密。加密后，只有收件人才能阅读加密的邮件。这样，在 Internet 上传递的电子邮件信息将不会被人窃取，即使发错邮件，收件人也无法看到邮件信息。
- 认证性：在 Internet 上传递电子邮件的双方互相不能见面，所以必须有方法确定发件人的身份。发件人在发信前对电子邮件进行数字签名，收信人便可据此确定发件人身份，而不是他人冒充的。
- 完整性：发件人对电子邮件进行数字签名不仅可确定发件人身份，而且可以保证发出的信息与接收到的完全一样，收信人能够通过验证数字签名来确保邮件完整性。
- 不可否认性：由于发件人对电子邮件进行的数字签名是唯一的，故发件人无法否认发过这个电子邮件。

9.1.2　安全电子邮件标准

邮件协议主要描述电子邮件发送端和接收端之间的传输过程以及定义邮件消息格式。与电子邮件有关的协议和标准主要有以下三个。

1. PEM(Privacy Enhanced Mail)增强型邮件保密标准

PEM 标准是由美国 RSA 实验室基于 RSA 和 DES 算法而开发的安全电子邮件的早期标准。PEM 在 Internet 电子邮件的标准格式上增加了加密、认证和密钥管理的功能。这个设计经过了几次更新，有关它的最后的更新可参阅 RFC1421-1424。由于 PME 依赖于一个即存的、完全可操作的 PKI，而建立一个符合 PEM 规范的 PKI 需要很长的过程，这些大大限制了 PEM 的发展，现在几乎已经没有人在用了。

2. MOSS(MIME Object Security Services)MIME 对象安全服务标准

MOSS 针对 PEM 的不足做了一些改进,它改变了 PEM 只支持文本信息的局面,支持了 MIME,将 PEM 和 MIME 两者的特性进行了结合,参见 RFC1848。但 MOSS 有很多的执行选项,这有可能导致两个不同的开发人员提出的两种 MOSS 邮件无法沟通。可以说,MOSS 往往被认为是一种框架而不是一个规范,在实现时还要考虑许多实际的问题,因此也没有被业界广泛使用。

3. PGP(Pretty Good Privacy)高质量保密标准

PGP 既是一个特定的安全电子邮件应用,也是一个安全电子邮件标准。PGP 符合 PEM 的绝大多数规范,但不必要求 PKI 的存在。它创造性地把 RSA 公钥体系的方便和传统加密体系的高速度结合起来,并在数字签名和密钥认证管理机制上有非常巧妙的设计。它不仅功能强大,速度很快,并且源代码公开。

4. S/MIME(Secure/Multipurpose Internet Mail Extension)安全/多用途因特网邮件扩展标准

S/MIME 是在 PEM 的基础上建立起来的,但是它发展的方向与 MOSS 不同。它选择使用 RSA 的 PKS(public-key cryptography standards)#7 标准,与 MIME 一起来保护电子邮件信息。S/MIME 已经成为产业界广泛认可的协议,如 Microsoft 公司、Netscape 公司、Novell 公司、Lotus 公司等都支持该协议。

MOSS 和 PEM 是没有被广泛实现的标准,目前的 Internet 上,有两套成型的端到端安全电子邮件标准:PGP 和 S/MIME。因此这里主要介绍 PGP 和 S/MIME。

9.2　电子邮件基本原理

一个电子邮件系统如图 9-1 所示,有两个主要组成部分,即用户接口和邮件传输程序。

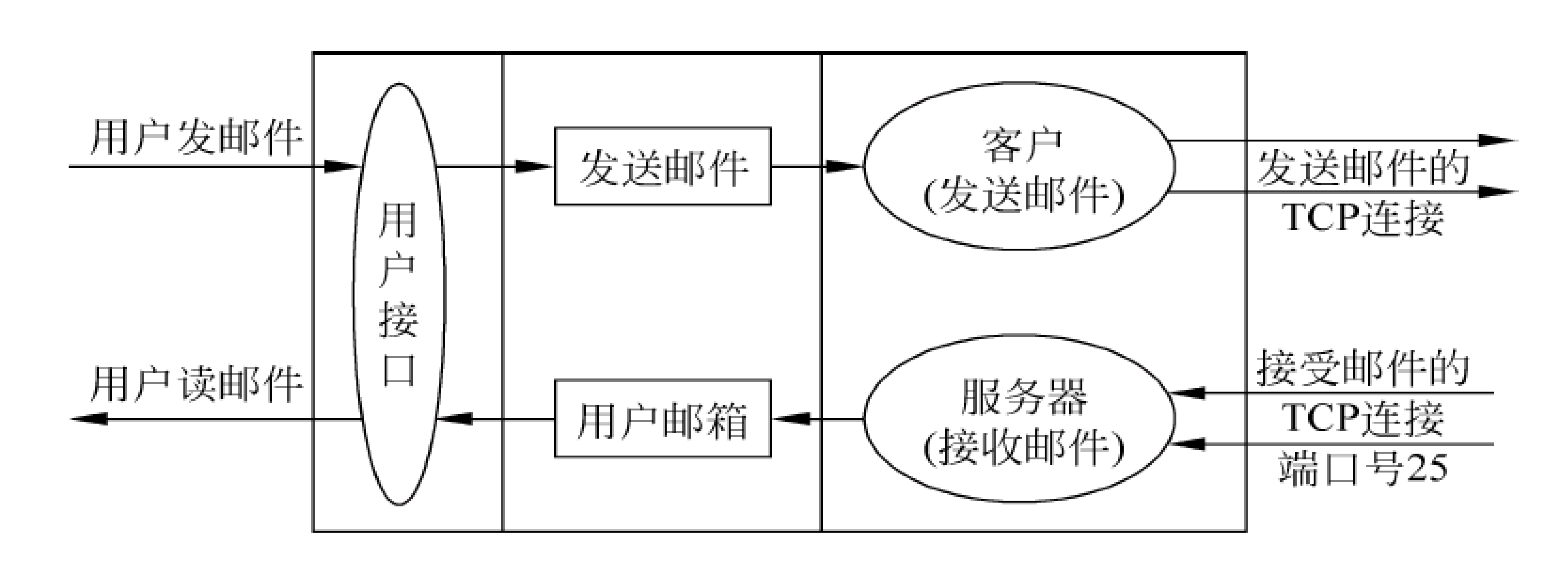


图 9-1　电子邮件系统

用户接口是一个在本地运行的程序,又称为用户代理(User Agent,UA),它使用户能够通过一个很友好的接口(目前主要是用窗口界面)来发送邮件和接收邮件。用户接口部分至少应当具有撰写、显示及处理邮件这三个功能。

邮件传输程序在后台运行,它将邮件通过网络发送给对方主机,并从网络接收邮件。邮件传输程序又称为消息传输代理(Message Transfer Agent,MTA),它具有传输、接收和报

告的功能。

电子邮件在传输过程中,往往需要经过多个节点。因此每一个网络节点都要安装邮件传输程序,即 MTA,以便对邮件进行存储转发。Internet 中的 MTA 的集合构成了消息传输系统(Message Transfer System,MTS)。

电子邮件传输机制如图 9-2 所示。

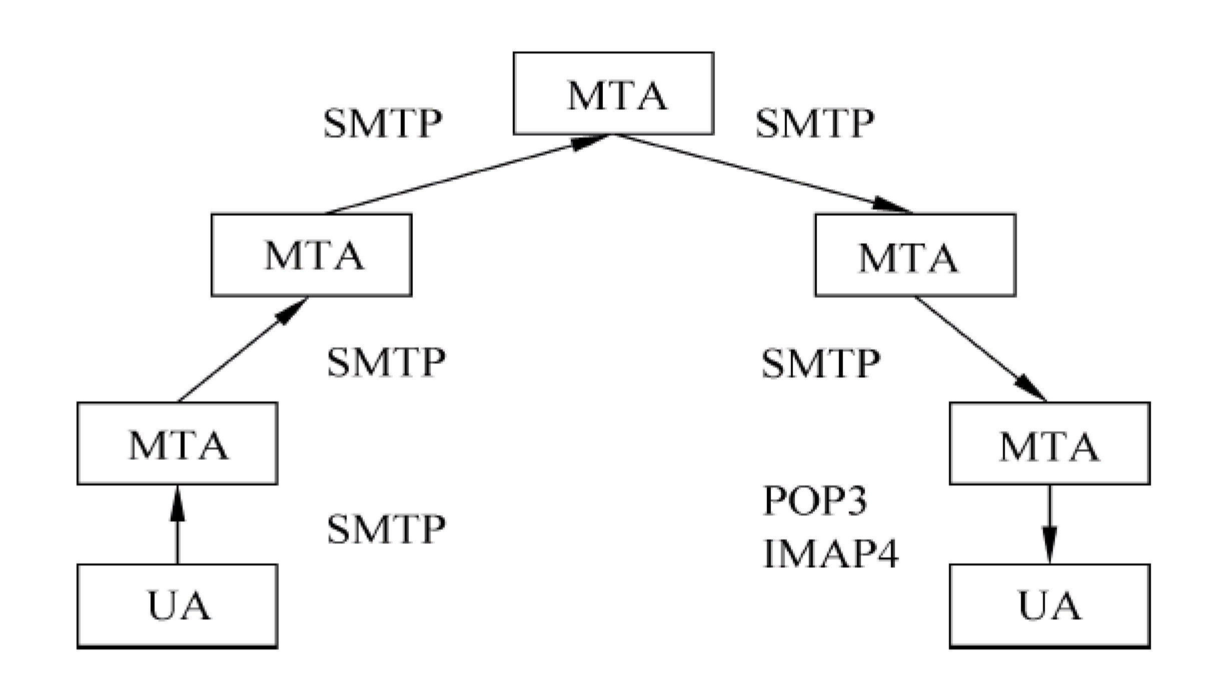


图 9-2 电子邮件传输机制

发送方用 UA 创建一个电子邮件,经本地的 MTA 发送,再经 MTS 传输至接收方的 MTA 进行接收,并使用接受者的 UA 显示出消息。

电子邮件系统的核心是 MTA。一个 MTA 通过 TCP 连接与另一个 MTA 进行通信,大部分 Internet 的 MTA 使用 SMTP 协议发送邮件、POP 协议收邮件并采用 MIME 进行编码。

9.3 电子邮件面临的威胁

电子邮件在 Internet 上是从一台机器传输到另一台机器。在这种方式下,电子邮件所经过的网络上的任一系统管理员都有可能截获或更改该邮件。目前电子邮件主要存在以下几种安全性问题。

1. 窃听

因为电子邮件需要在网络上传播,所以很容易成为窃听的目标。根据电子邮件的传输机制,进行窃听是很容易的,窃听不仅可以出现在局域网,在客户机和服务器之间的所有点上都有可能存在。邮件内容和信封的内容都可以被窃听。

2. 假冒

SMTP 不提供任何验证机制,所以伪造和篡改电子邮件是很容易的。伪造者只要给 SMTP 服务器提供合适的信封信息(如发送方或接收方电子邮件地址),并使用想要的数据产生有关的信件即可。与假冒相关的另一个安全问题是抵赖问题,因为信件可以伪造,发送方可以否认他曾经发送过某个信件。

3. 拒绝服务攻击

这种攻击通过破坏机器之间的网络通信，或者是通过耗用邮件服务器上的资源，如磁盘空间、内存等实现。

4. 恶意邮件

例如邮件附件中附带有病毒、蠕虫、木马程序以及大量的垃圾邮件，会对接收方的计算机系统和整个计算机网络带来严重破坏。

9.4　PGP

PGP(Pretty Good Privacy)是一个基于 RSA 公钥密码体系的邮件安全软件。可以用它对邮件加密以防止非授权者阅读，对邮件进行数字签名从而使收信人可以确认邮件的发送者，并能确信邮件没有被篡改。它可以提供一种安全的通信方式，而事先并不需要任何保密的渠道用来传递密钥，而且它的源代码是免费的。

9.4.1　PGP 概述

PGP 主要是 Zimmermann 一个人的贡献，PGP 提供了一种机密性和鉴别的服务，可以用于电子邮件和文件存储的应用。Zimmermann 主要做了下面一些工作：

(1) 选择最好的、可用的加密算法作为基础构件；

(2) 将这些算法集成起来，形成一个通用的独立于操作系统和处理器的应用程序，该应用程序基于易于使用的命令构成的小集合；

(3) 制作了软件包和相应的文档，包括源代码，通过 Internet、电子公告牌和如 Compuserve 的商业网，免费提供给用户；

(4) 和公司签订了合同，提供 PGP 的完全兼容的、低价格的商用版本。

PGP 发展非常迅速，现在已经被广泛应用。PGP 得到快速发展有以下一些原因。

(1) 可免费得到，包括运行在不同平台的多个版本，诸如：DOS/Windows，UNIX，Macintosh 等。另外，商用版本满足了那些想要获得厂家技术支持的用户的需要。

(2) 建立在一些经过公开评议，被认为是非常安全的算法的基础上。具体来说，PGP 软件包包括了公开密钥的 RSA，DSS 和 Diffie-Hellman 加密算法；常规的 CAST-128，IDEA 和 3DES 加密算法；以及散列编码的 SHA-1 算法。

(3) PGP 的应用范围非常广，从想要选择和实施一个标准模式来加密文件和消息的公司到想要通过 Internet 或其他网络和他人进行安全通信的个人都可以使用。

(4) 不是任何政府或标准化组织开发的，因而也不会被它们控制。对于那些本质上不信任任何“组织机构”的人来说，PGP 很具有吸引力。

9.4.2　PGP 提供的安全服务

与密钥的管理相比，PGP 的实际操作由五种服务组成：鉴别、机密性、压缩、电子邮件的兼容性和分段(见表 9-1)。

表 9-1　PGP 服务一览表

功　　能	使用的算法	描　　述
数字签名	DSS/SHA 或 RSA/SHA	使用 SHA-1 创建的消息的散列编码。采用 DSS 或 RSA 算法使用发送方的私有密钥对这个消息摘要进行签名,并且包含在消息中
消息加密	CAST 或 IDEA 或 3DES,带有 Diffie-Hellman 算法或 RSA	采用 CAST-128 或 IDEA 或 3DES,使用发送方生成的一次性会话密钥对消息进行加密,采用 Diffie-Hellman 或 RSA,使用接收方的公开密钥对会话密钥进行加密并包含在消息中
压缩	ZIP	消息可以使用 ZIP 进行压缩,用于存储或传输
电子邮件兼容性	64 基转换	为了提供电子邮件应用的透明性,加密的消息可以使用 64 基转换算法转换成 ASCII 字符串
分段	—	为了满足最大消息长度的限制,PGP 完成消息的分段和重新装配

下面将详细介绍这五种服务。

首先,介绍将要使用的符号。

K_s:session key;

KR_a:用户 A 的私钥;

KU_a:用户 A 的公钥;

EP:公钥加密机制;

DP:公钥解密机制;

EC:常规加密;

DC:常规解密;

H:散列函数;

||:连接符;

Z:用 ZIP 算法压缩;

$R64$:用 radix64 转换到 ASCII 格式;

M:信件。

1. 鉴别

图 9-3 显示了 PGP 提供的数字签名服务。

图 9-3 显示的这些步骤,描述如下:

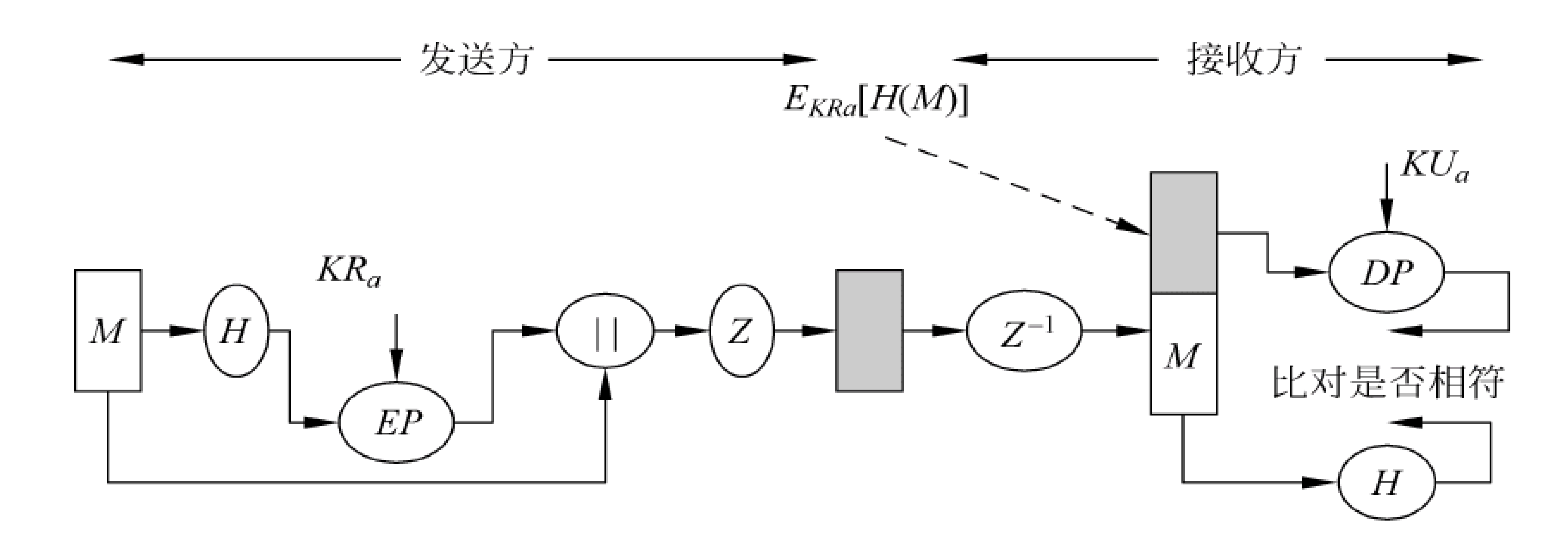


图 9-3　PGP 数字签名

发送方创建消息 M。

① 发送方一端的 PGP 机制用 SHA-1、MD5 等 Hash 函数来生成消息的 160b 散列代码。

② 使用发送方的私有密钥,采用 RSA 算法对散列代码进行加密(数字签名),结果串接在消息的前面一起发送到接收方一端。

③ 接收方收到传送的消息后，使用发送方的公开密钥，采用 RSA 解密和恢复散列代码。

④ 接收方将收到的消息生成新的散列代码，并与被解密的散列代码相比较。如果两者匹配，则消息作为已鉴别的消息而接受。

SHA-1 与 RSA 的结合提供了一种有效的数字签名模式。由于 RSA 的强度，接收方可以确认只有私有密钥的拥有者才能生成该签名。由于 SHA-1 的安全强度，接收方可以相信没有其他人能生成新消息，该消息能够匹配散列代码，因而匹配原始消息的签名。另外，签名也可以使用 DSS/SHA-1 作为可选方案来生成。

一般来说，签名和所签的消息或文件附加在一起，但分离的签名也是允许的。例如用户可能想要维护一个单独的签名日志，其中包括所有发送和接收的消息；另外，可执行程序的分离签名可以检测出随后的病毒侵入。在需要多方签名的文档上，如法律合同，可以用分离签名，每个人的签名是独立的，因而可以只应用于文档。否则，签名将只能递归使用，第二个签名对文档和第一个签名进行签名，依此类推。

2. 机密性

PGP 通过对消息进行加密来保证消息的机密性。加密的方法可以分为常规的对称密钥加密和公钥加密，在 PGP 提供机密性服务的过程中，两者都必须应用。在 PGP 中，每个会话密钥只使用一次，即对每个消息生成新的 128b 的随机数。因为在文档中会话密钥是一次性密钥，只被使用一次，所以会话密钥与消息绑定在一起并和消息一起传输。为了保护密钥，使用接收方的公开密钥对它进行加密。图 9-4 显示了这一步骤，描述如下：

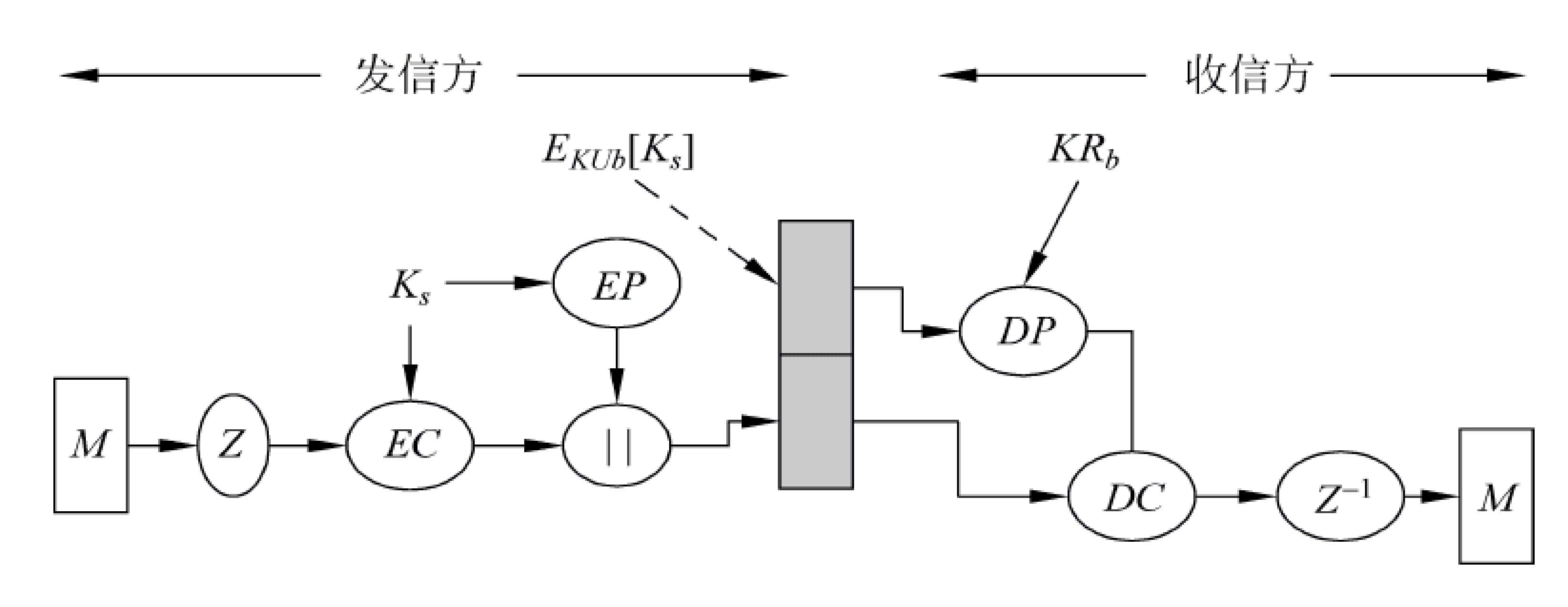


图 9-4　PGP 加密

① 发送方生成消息并使用随机数生成器 RNG 产生 128 b 的对称密钥加密所需的会话密钥。

② 采用 CAST-128（或 IDEA 或 3DES）算法，使用生成的会话密钥对消息进行加密。

③ 采用 RSA 算法，使用接收方的公开密钥对会话密钥进行加密，并附加到消息前面。

④ 接收方采用 RSA 算法，使用自己的私有密钥解密得到会话密钥。

⑤ 使用解密得到的会话密钥来解密消息。

除了使用 RSA 对密钥加密之外，PGP 提供了另一种称为 Diffie-Hellman 的密钥交换算法。实际上，PGP 使用的是 Diffie-Hellman 的提供加密/解密的变体。CAST-128 和其他常规算法比 RSA 要快得多，为了减少加密的时间，对称密钥加密和公钥加密结合使用比直接使用 RSA 来加密消息更好。使用公开密钥算法解决了会话密钥分配问题，因为只有接收方才能恢复绑定到消息上的会话密钥。考虑到电子邮件的存储转发特性，使用握手协议来保

证双方具有相同会话密钥的方法是不现实的。相反,每个消息都是一次性的,带有自己密钥的独立事件。使用一次性的常规密钥加强了已经是很强的常规加密方法。

3. 机密性与鉴别

如图 9-5 所示,对消息可以同时使用机密性与鉴别两个服务。首先,为明文消息生成签名并附加到消息首部。然后使用 CAST-128(或 IDEA 或 3DES)对明文消息和签名进行加密,再使用 RSA 对会话密钥进行加密。在这里要注意次序,如果先加密再签名的话,别人可以将签名去掉后签上自己的签名,从而篡改签名。

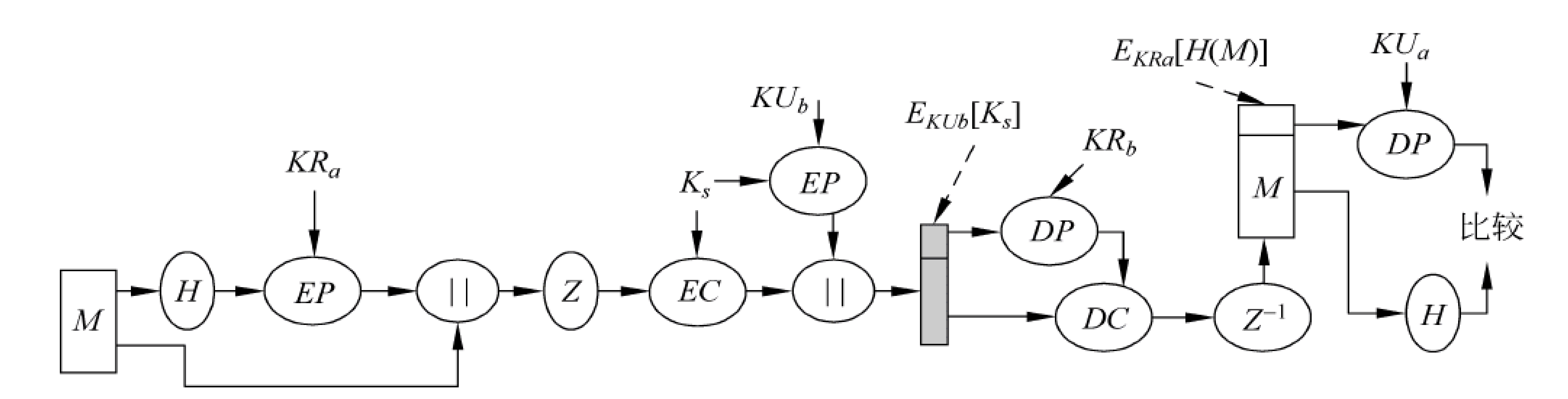


图 9-5 PGP 加密并签名

总之,当同时使用机密性和鉴别两个服务时,发送方首先用自己的私有密钥为消息签名,然后使用会话密钥对消息加密,接着再用接收方的公开密钥对会话密钥进行加密。

4. 压缩

PGP 在加密前进行预压缩处理,PGP 内核使用 PKZIP 算法压缩加密前的明文。一方面对电子邮件而言,压缩后再经过 radix-64 编码有可能比明文更短,这就节省了网络传输的时间和存储空间;另一方面,明文经过压缩,实际上相当于经过一次变换,对明文攻击的抵御能力更强。

5. 电子邮件的兼容性

当使用 PGP 时,至少传输消息的一部分需要被加密。如果只使用签名服务,那么消息摘要被加密(使用发送方的私有密码)。如果使用机密性服务,消息加上签名(如果存在)被加密(使用一次性的对称密码)。因此,部分或全部的结果消息由任意的 8b 字节流组成。但是,很多电子邮件系统只允许使用由 ASCII 正文组成的块。为了满足这一约束,PGP 提供了将原始 8b 二进制流转换成可打印 ASCII 字符的服务。

为这一目的采用的方案是 radix-64(也是基数 64)转换,每三个字节的二进制数据为一组映射成四个 ASCII 字符。这种格式还附加了 CRC 校验来检测传输错误。

使用 radix-64 将消息的长度扩充了 33%。幸运的是消息的会话密钥和签名部分相对紧凑,并且明文消息已经进行了压缩。实际上压缩足以补偿 radix-64 的扩展。

radix-64 算法的一个值得注意的方面是它盲目地将输入流转换成 radix-64 的格式,而不管输入流的内容,即使输入流正好是 ASCII 正文。因此,如果消息被签名但还没有被加密,并且对整个分组应用上述转换,输出的结果对于偶然的观察者是不可读的,这提供了一定程度的机密性。作为选项,PGP 可以配置成只将签名的明文消息的签名部分转换成 radix-64 的格式,这可以使得接收消息的人不使用 PGP 就能阅读消息。当然,必须使用 PGP 才能验证签名。

6. 分段和重装

电子邮件设施经常受限于最大的消息长度。例如，很多 Internet 可以访问的设施都有最大 50000 个八位组的限制。任何长度超过这个数值的消息都必须划分成更小的消息段，每个段单独发送。

为了满足这个约束，PGP 自动将太长的消息划分成可以使用电子邮件发送的足够小的消息段。分段是在所有其他的处理（包括 radix-64 转换）完成之后才进行的，因此会话密钥部分和签名部分只在第一个消息段的开始位置出现一次。在接收端，PGP 必须在完成图 9-5 所显示的步骤之前剥掉所有的电子邮件首部，并且重新装配成完整的原来的分组。

9.4.3 PGP 消息格式及收发过程

消息由三个部分组成：消息部分、签名（可选）和会话密钥部分（可选），如图 9-6 所示。

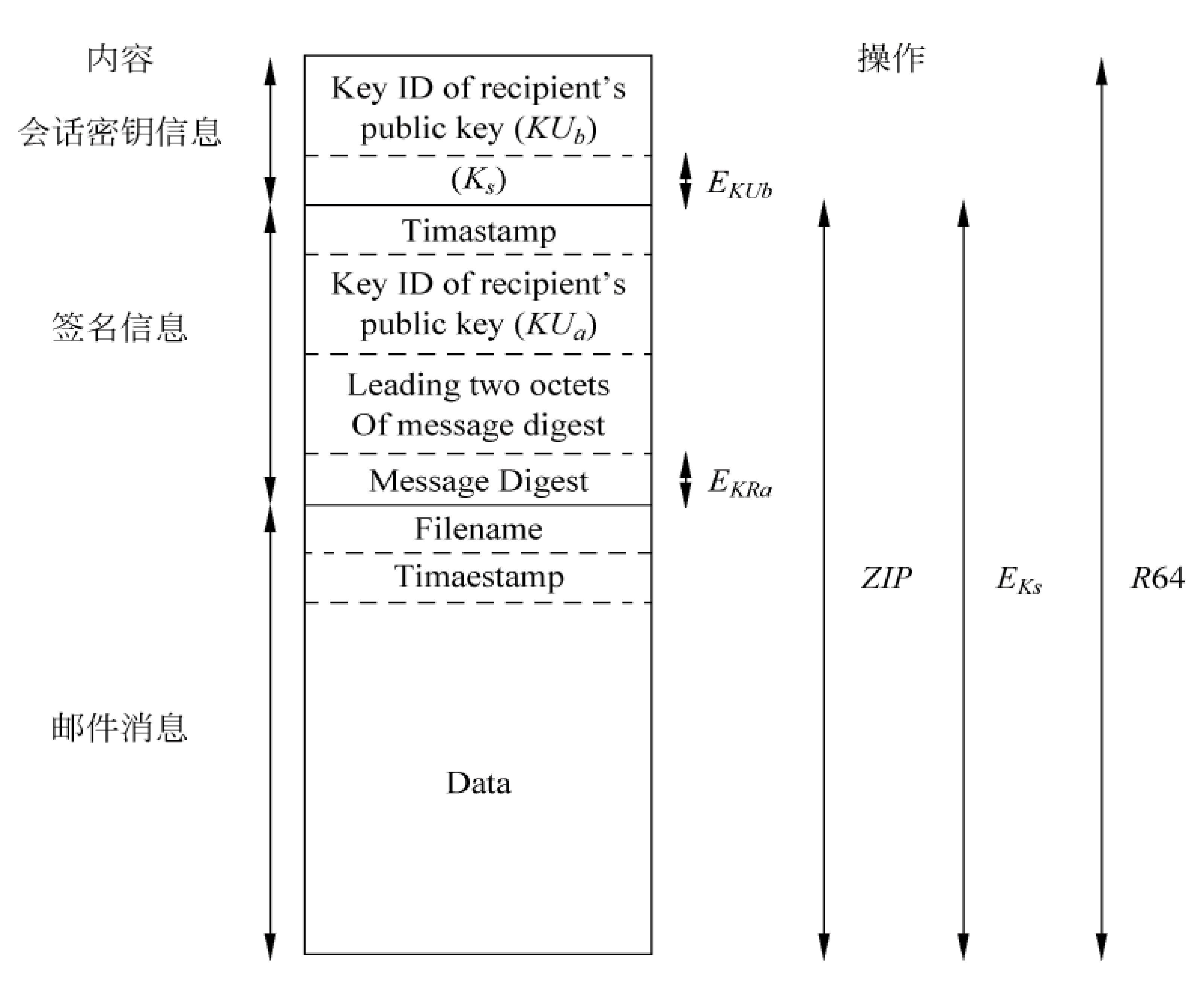


图 9-6　PGP 消息格式

其中 E 表示进行加密；ZIP 表示进行压缩；$R64$ 表示进行 64 基转换。

会话密钥信息包括：会话密钥，以及发送方用来对会话密钥加密的接收方的公开密钥的标识符。

签名部分包括下面一些成分。

① 时间戳：签名构造的时间。

② 消息摘要：160b 的 SHA-1 摘要，使用发送方的私有签名密钥进行加密。摘要从签名时间戳算起一直到消息部分的数据段，在摘要中包括签名时间戳是为了防止重放类型的攻击。排除了消息部分的文件名和时间戳段是为了保证分离的签名和附加在消息前面的签名完全一致，分离的签名是在单独的文件上计算的，该文件没有任何消息部分首部的字段。

③ 消息摘要的前两个字节：通过将这两个字节的明文与加密摘要的前两个字节相比

较,使得接收方可以决定是否使用了正确的公开密钥。这些八位组还作为消息的16b帧检验序列。

④ 发送方公开密钥的密钥ID:标识了应该用来解密消息摘要的公开密钥,因此,标识了用来加密消息摘要的私有密钥。

消息部分包括了文件名、说明创建时间的时间戳以及要存储或传输的实际的数据。

下面介绍PGP的消息收发过程。

1. 消息发送

消息发送如图9-7所示。

在传送时,如果需要,使用明文的散列编码生成签名。然后,一起压缩明文和签名(如果存在)。接下来,如果需要机密性,则把压缩过的明文(加签名)分组加密,并且在前面附加公开密钥加密过的常规加密密钥。最后,这个分组被转换成radix-64的格式。

2. 消息接收处理

消息接收处理如图9-8所示。

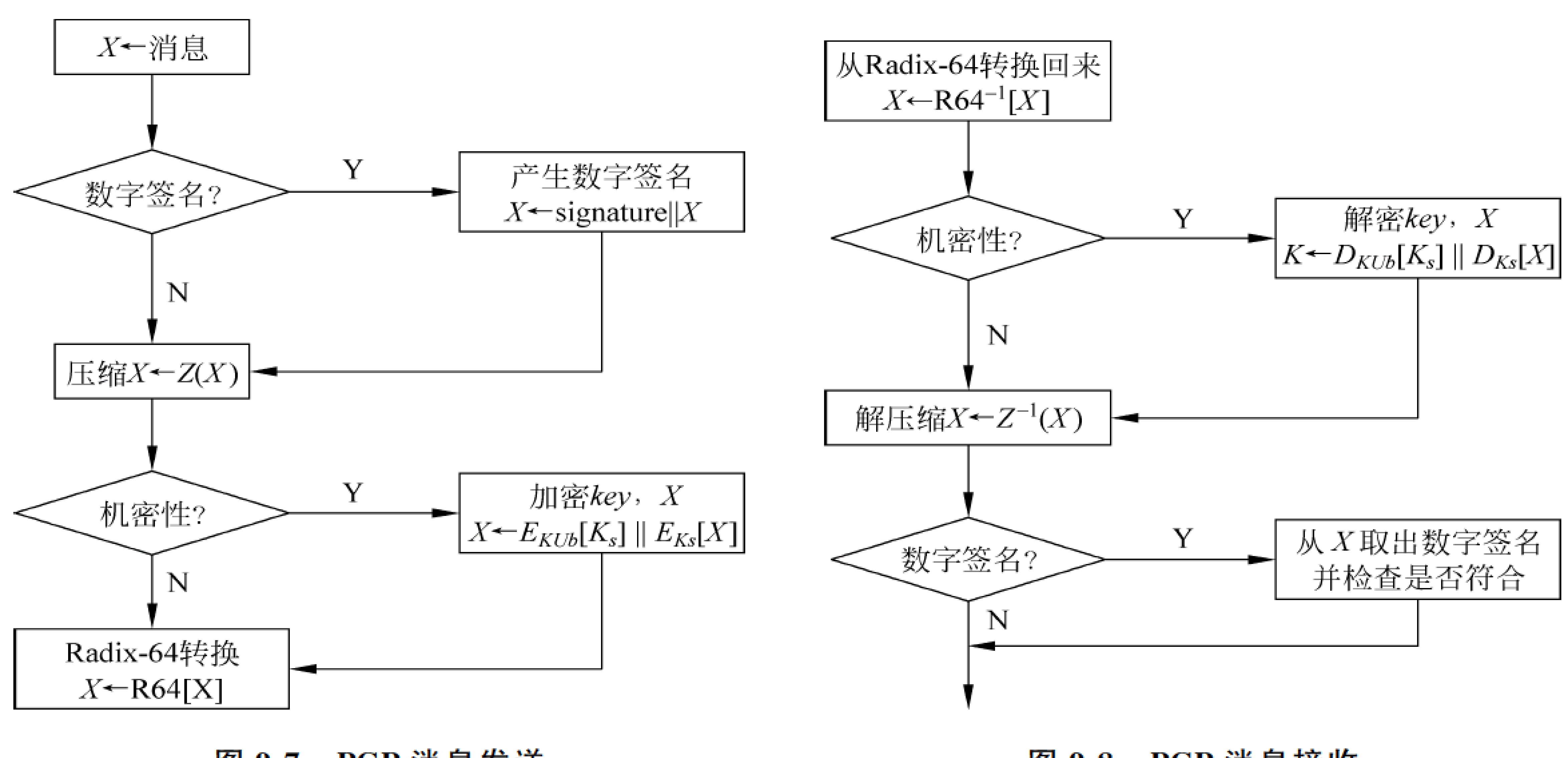


图9-7　PGP消息发送　　　　**图9-8　PGP消息接收**

在接收时,进入的分组首先从radix-64转换成二进制形式。然后,如果消息被加过密,接收方恢复会话密钥并解密消息。接着,对结果消息解压。如果消息签了名,接收方恢复传输过来的散列编码,并与自己计算出来的散列编码进行比较。

9.4.4　PGP的密钥管理

PGP密钥管理主要分为以下几个部分:加密密钥和密钥环,会话密钥的生成,密钥环,公钥环,公开密钥管理,信任的使用,撤销公钥。

1. 加密密钥和密钥环

PGP利用了四种类型的密钥:一次性会话的常规密钥、公开密钥、私有密钥和基于口令短语(passphrase)的常规密钥(在下面解释)。对于这些密钥存在以下需求:

（1）需要生成一种不可预测的会话密钥的方法。

（2）允许用户拥有多个公开/私有密钥对。

理由是用户可能想要不时地改变他或她的密钥对。当发生了这种情况时，管道中所有消息都将是用过时的密钥构造的。此外，接收方在一个更新到达之前只知道旧的公开密钥。为了满足不时地改变密钥的需要，在给定的时刻，用户可能想要拥有多个密钥对来和不同的通信组交互，或者只是为了增强安全性而限制密钥能够加密的消息的数量。所以在用户和他们的公开密钥之间不存在一一对应的关系。因此，需要某种方法来标识特定的密钥。

（3）每个 PGP 实体必须维护它自己的公开/私有密钥对的文件，以及与它通信者的公开密钥文件。

2. 会话密钥的生成

每个会话密钥都和一个消息关联，并且只是为了加密和解密该消息而使用。消息的加密和解密是采用对称的加密算法来完成的。CAST-128 和 IDEA 使用 128b 密钥；3DES 使用 168b 的密钥。在下面的讨论中假定使用 CAST-128。

随机的 128b 数是使用 CAST-128 本身来生成的。随机数生成器的输入由 128b 密钥和两个看做是要加密的明文的 64b 分组组成的。使用加密反馈模式，CAST-128 加密器产生两个 64b 密码正文分组，这两个分组串接起来形成了 128b 会话密钥。PGP 的会话密钥是个随机数，它是基于 ANSIX12.17 的算法由随机数生成器产生的。随机数生成器从用户敲键盘的时间间隔上取得随机数种子。所以这里使用的算法是基于 ANSIX12.17 所描述的算法。

随机数生成器的两个 64b 分组组成的“明文”输入本身也是从 128b 随机化的数值流中推导而来的，这些数值依赖于用户的击键输入。击键的时间和输入的字符被用来生成随机化的数值流，因此，如果用户以正常速度击了任意键，将会生成合理的“随机”输入。这个随机输入和前面 CAST-128 输出的会话密钥组合在一起输入到生成器中。考虑到 CAST-128 有效的不规则性，产生的结果就是不可预测的会话密钥的序列。

前面提到，加密消息伴随着加密形式使用会话密钥。会话密钥本身使用接收方的公开密钥进行加密，因此只有接收方才能恢复会话密钥，因而能够恢复消息。如果用户配置了单个的公开/私有密钥对，那么接收方将自动知道使用哪个私有密钥来解密会话密钥。然而，前面说过存在给定用户拥有多个公开/私有密钥对的需求。那么，接收方如何知道消息使用了它的哪个公开密钥来加密的呢？

一种简单的解决方法是将公开密钥与消息一起传输。接收方可以验证该公开密钥确实是它的公开密钥，然后进一步处理。这种方法能够解决问题，但 RSA 公开密钥的长度可能是几百个十进制数，这样就没有必要地浪费了空间。另一种解决方法是给每个公开密钥指派至少在一个用户内部唯一的标识符，也就是将用户 ID 和密钥 ID 组合在一起来唯一标识一个密钥，那么只有小得多的密钥 ID 需要被传输。但是，密钥 ID 必须被赋值和存储，使得发送方和接收方都能够将密钥 ID 映射成公开密钥，这带来了管理和负载上的问题。

PGP 采用的解决方法是给每个公开密钥指派一个密钥 ID，这在用户 ID 中很可能是唯一的。密钥 ID 与每个公开密钥相关联，并由公开密钥的最低 64 比特组成，即公开密钥 KU_a 的密钥 ID 为 $KU_a \bmod 2^{64}$。这个长度足以使密钥 ID 重复概率非常小。

PGP 的数字签名也需要密钥 ID。因为发送方可能使用一组私有密钥中的一个来加密

消息摘要,接收方必须知道应该使用哪个公开密钥来解密。相应地,消息的数字签名部分包括了需要的公开密钥的 64b 密钥 ID。当消息被收到时,接收方可以验证该密钥 ID 是它所知道的发送方的公开密钥,然后接着来验证签名。

3. 密钥环

前面介绍了密钥 ID 对于 PGP 的操作是如何重要,任何一个 PGP 消息中包括的两个密钥 ID 提供了机密性和鉴别。这些密钥需要以一种系统化的方法来存储和组织,使得各方能够有效和高效地使用它们。PGP 使用的方法是在每个结点提供一对数据结构,一个用来存储该结点拥有的公开/私有密钥对,另一个用来存储该结点所知道的其他用户的公开密钥。相应地,这些数据结构被称为私有密钥环和公开密钥环。

可以把私有密钥环看做是一张表,见表 9-2。

表 9-2 私有密钥环的一般结构

时间戳	密钥 ID	公钥	加密的公钥	用户 ID
…	…	…	…	…
T_i	$KU_i \bmod 2^{64}$	KU_i	$E_{H(P_i)}[KR_i]$	T_i
…	…	…	…	…

其中每一行代表了该用户拥有的公开/私有密钥对。每一行包含了下面一些实体。

- 时间戳:该密钥对生成的日期/时间。
- 密钥 ID:这个实体的公开密钥的低位 64 比特。
- 公开密钥:密钥对的公开密钥部分。
- 私有密钥:密钥对的私有密钥部分;这个字段是加密的。
- 用户 ID:这个字段的典型值是用户的电子邮件地址。然而,用户可以为每个密钥对选择不同的名字,或者多次重用相同的用户 ID。

私有密钥环可以用用户 ID 或密钥 ID 来索引,后面将介绍创建这两种索引的情况。

尽管想要私有密钥环只存储在创建和拥有密钥对的用户的机器上,并且只有该用户可以访问私有密钥环,但确保私有密钥的值尽可能地安全仍然很有意义。相应地,私有密钥本身并不存储在密钥环中,相反,要使用 CAST-128(或 IDEA 或 3DES)来对这个密钥加密。其步骤如下:

(1) 用户选择口令短语用来对私有密钥进行加密。

(2) 当系统使用 RSA 生成新的公开/私有密钥对时,它向用户要求该口令短语。使用 SHA-1 生成该口令短语的 160b 的散列编码,然后该口令短语被丢弃。

(3) 系统使用 CAST-128,将 128b 的散列编码作为密钥来加密私有密钥。然后,散列编码被丢弃,加密过的私有密钥被存储在私有密钥环中。

接下来,当用户访问私有密钥环来查询私有密钥时,他或她必须提供相应的口令短语。PGP 将查询加密的私有密钥,生成口令短语的散列编码,然后使用该散列编码来应用 CAST-128 对加密的私有密钥进行解密。

这是非常简洁和有效的方法。和任何基于口令的系统一样,系统的安全性依赖于口令的安全性。为了避免将口令写下来,用户应该使用一种不容易猜但却容易记住的口令短语。

4. 公钥环

表 9-3 显示了公开密钥环的一般结构。这个数据结构用来存储该用户知道的其他用户的公开密钥。表中实体含义和私钥环中含义基本相同，也可以用用户 ID 或密钥 ID 来索引。

表 9-3　公钥环的一般结构

时间戳	密钥 ID	公钥	拥有信任者	用户 ID	密钥合理性	签名	签名信任
…	…	…	…	…	…	…	…
T_i	$KU_i \bmod 2^{64}$	KU_i	$Trust_flag_i$	$User_i$	$Trust_flag_i$		
…	…	…	…	…	…	…	…

下面介绍这些密钥环是如何用于消息的传输和接收的。为了简单起见，在以下的讨论中忽略压缩和 radix-64 转换。

首先考虑消息传输，假定消息将既被签名又被加密，如图 9-9 所示。

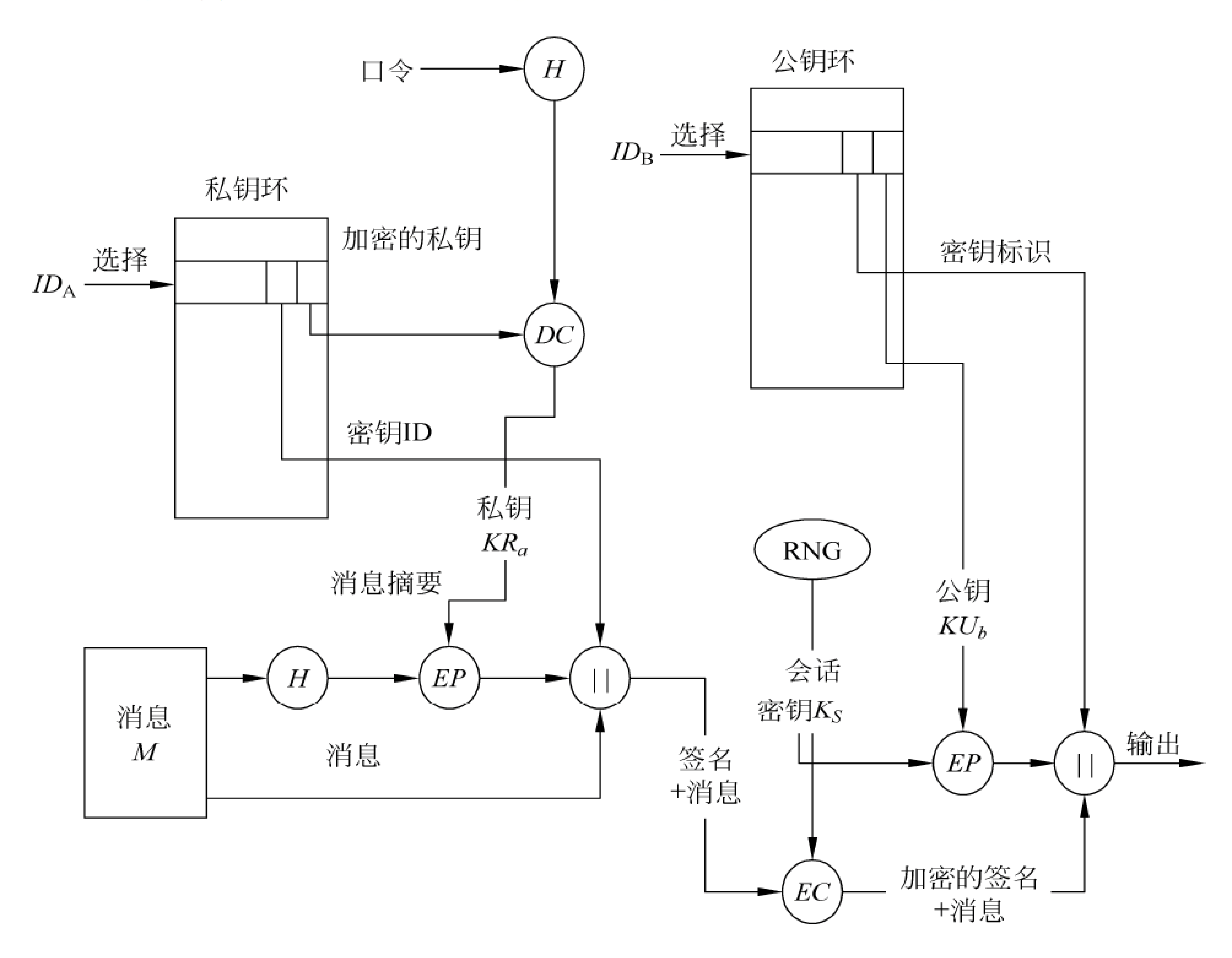


图 9-9　PGP 发送端工作流程

发送 PGP 实体完成下面一些动作。

(1) 签名消息。PGP 使用用户 ID 作为索引从私有密钥环中查找发送方的私有密钥。如果在命令中没有提供用户 ID，则使用环中的第一个私有密钥。提示用户输入口令短语来恢复未加密的私有密钥。构造消息的签名部分。

(2) 加密消息。PGP 生成会话密钥并且加密消息。使用接收方的用户 ID 作为索引，从公开密钥环中查找接收方的公开密钥。构造消息的会话密钥部分。

接收 PGP 实体完成下面一些步骤(如图 9-10 所示)。

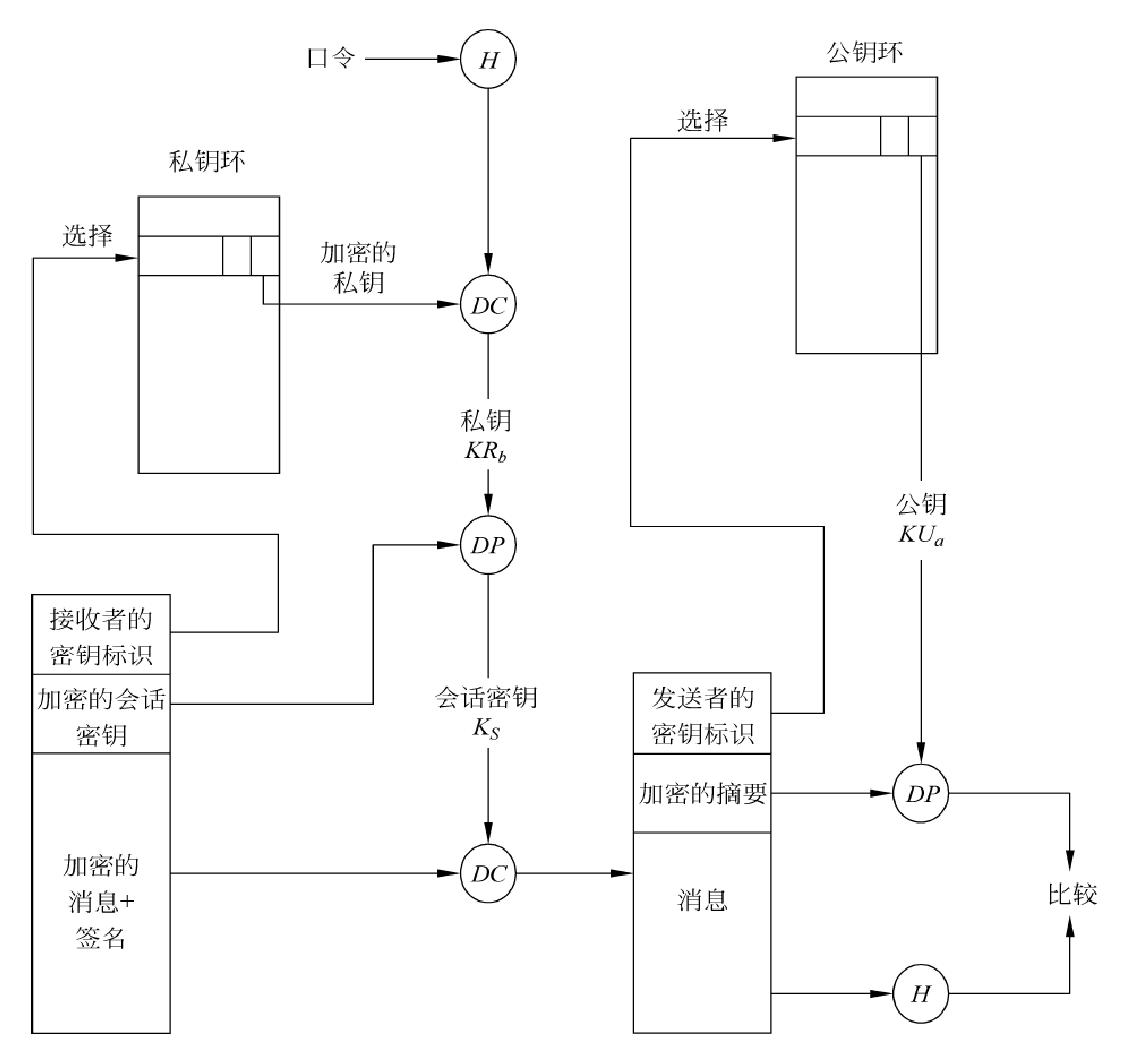


图 9-10 PGP 接收端工作流程

(1) 解密消息。PGP 使用消息会话密钥部分的密钥 ID 字段作为索引,从私有密钥环中查找接收方的私有密钥。提示用户输入口令短语来恢复未加密的私有密钥。然后恢复会话密钥并且解密消息。

(2) 鉴别消息。PGP 使用消息签名密钥部分的密钥 ID 字段作为索引,从公开密钥环中查找发送方的公开密钥,恢复传输的消息摘要。对接收的消息计算其消息摘要,将计算的结果与传输过来的消息摘要相比较以鉴别消息。

5. 公开密钥管理

保护公开密钥不被篡改在公开密钥应用中是最困难的问题。它是公开密钥加密方法中唯一的薄弱环节,很多的软件复杂性都是为了解决这个问题而产生的。

PGP 提供了解决这个问题的结构和一些可以使用的选项。因为 PGP 要用在多种正式或非正式的环境中,因此并没有建立严格的公开密钥管理方法。

以用户 A 和用户 B 通信为例,现假设用户 A 给用户 B 发信,首先用户 A 的公钥环就必须包含用户 B 的公钥,用户 A 得到用户 B 的公钥,并把用它加密的信件发给用户 B。但是用户 A 和用户 B 都不知道,另一个用户 C 侦听或截取到用户 B 的公钥,然后用自己的公钥替换了用户 B 的公钥,并发给用户 A。这样一来用户 B 收到用户 A 的来信后就不能用自己

的私钥解密了，用户 C 还可伪造用户 B 的签名给用户 A 或其他人发信，因为用户 A 手中的公钥是伪造，用户 A 会以为真是用户 B 的来信，用户 A 发给用户 B 的加密消息都可以被用户 C 看到。

为了把公开密钥的危险最小化，用户 A 可以采取以下方法来获得用户 B 的可靠公钥。

（1）直接从用户 B 手中得到她的公钥，然而当她远在他乡或在时间上根本不可达到时，这是不可办到的。

（2）通过电话验证密钥。如果用户 A 可以在电话上听出用户 B 的声音，用户 A 可以给用户 B 打电话，要求她在电话上以 radix-64 的形式口述密钥。更加实用的方法是用户 B 可以将她的密钥通过电子邮件传递给用户 A。然后，用户 A 可以让 PGP 生成密钥的 160b 的 SHA-1 摘要，并且以十六进制的形式显示，这个摘要被称为密钥的“指纹”。接下来，用户 A 可以给用户 B 打电话，让她在电话里口述“指纹”。如果两个“指纹”一致，那么密钥就得到了验证。

（3）从双方信任的第三方 D 那里获得用户 B 的公开密钥。介绍人 D 创建签名的证书，证书包括用户 B 的公开密钥、创建密钥的时间和密钥的合法期限。介绍人 D 生成这个证书的 SHA-1 摘要，使用他的私有密钥对其加密，并且将签名附加在证书上。因为只有 D 能够创建这个签名，所以没有其他人可以创建假的公开密钥，并且伪装成经过 D 的签名。签名的证书可以直接由用户 B 或第三方 D 发送给用户 A，或者公布在公告牌上。

（4）从可信任的认证中心获得用户 B 的公开密钥。再一次创建公开密钥证书，并经过认证中心的签名。用户 A 可以访问这个认证中心，提供用户名获得签名的证书。

对于第（3）和第（4）种情况，用户 A 应该已经有了介绍人的公开密钥，并且可以相信这个密钥是合法的。最终来说，用户 A 必须对承担介绍人角色的人指派一定程度的信任。

6. 信任的使用

PGP 确实为公开密钥附加信任和开发信任信息提供了一种方便的方法使用信任。公开密钥环的每个实体都是一个公开的密钥证书。与每个这样的实体相联系的是密钥合法性字段，用来指示 PGP 信任“这是这个用户合法的公开密钥”的程度越高，这个用户 ID 与这个密钥的绑定越紧密。这个字段由 PGP 计算。与每个实体相联系的还有用户收集的多个签名。反过来，每个签名都带有签名信任字段，用来指示该 PGP 用户信任签名者对这个公开密钥证明的程度。密钥合法性字段是从这个实体的一组签名信任字节中推导出来的。最后，每个实体定义了与特定的拥有者相联系的公开密钥，包括拥有者信任字段，用来指示这个公开密钥对其他公开密钥证书进行签名的信任程度（这个信任程度是由该用户指定的）。可以把签名信任字段看成是来自于其他实体的拥有者信任字段的副本。

例如正在处理用户的公开密钥环，操作描述如下：

（1）当 A 在公开密钥环中插入了新的公开密钥时，PGP 必须为与这个公开密钥拥有者相关联的信任标志赋值。如果拥有者是 A，因此这个公开密钥也出现在私有密钥环中，那么这个信任字段被自动指派终极信任的值。否则，PGP 询问 A 指派这个密钥拥有者的信任程度的评定，A 必须输入想要的级别。用户可以说明这个拥有者是未知的、不可信任的、少量信任和完全可信的。

（2）当新的公开密钥输入之后，可以在它上面附加一个或多个签名，以后还可以增加更多的签名。在实体中插入签名时，PGP 在公开密钥环中搜索，查看这个签名的作者是否属于已知的公开密钥拥有者。如果是，为这个签名的 SIGTRUST 字段指派该拥有者的

OWNERTRUST 值。否则,指派不认识的用户值。

(3) 密钥合法性字段的值是在这个实体的签名信任字段的基础上计算的。如果至少有一个签名具有终极信任的值,那么密钥合法性字段的值设置为完全;否则,PGP 计算信任值的权重和。对于总是可信任的签名指派 $1/X$ 的权重,对于通常可信任的签名指派 $1/Y$ 的权重,其中 X 和 Y 都是用户可配置的参数。当介绍者的密钥/用户 ID 绑定的权重总和达到 1 时,绑定被认为是值得信赖的,密钥合法性值被设置为完全。因此,在没有终极信任的情况下,需要至少 X 个签名是总是可信的,或者至少 Y 个签名是通常可信的,或者上述两种情况的某种组合。

7. 撤销公钥

用户可能想要废除当前的公开密钥,废除公开密钥的常规方法是拥有者发出密钥废除证书,该证书必须由拥有者用其私有密钥签名。这个证书和一般的签名证书的格式一样,但其中包含一个标志证书的目的是撤销其公钥的使用。拥有者应该尽可能快速、广泛地散布这个证书,使得用户都能够更新他们的公开密钥环。虽然窃取此私有密钥的敌手也可以发出这样的撤销证书。但这会使得这个敌手和合法的拥有者一样不能使用该公开密钥,因此这看起来比恶意使用被偷的私有密钥的威胁要小得多。

9.4.5 PGP 应用

PGP Desktop 是一个安全工具,它使用加密技术来保护用户的数据,使其免遭未经授权的访问。

当发送电子邮件或即时消息(IM)时,PGP Desktop 保护用户的数据。它可以让用户加密自己的整个硬盘或硬盘分区,因此所有的内容可以一起被保护起来。通过虚拟磁盘,用户可以安全地存储最敏感的数据。用户也可以使用它在网络上安全地分享文件和文件夹。它可以让用户把任何组合的文件和文件夹放到一个加密的压缩包中,以方便分配或备份。PGP Desktop 还可以让用户安全删除敏感文件和硬盘空间,因此没有人可以找回它们,也没有任何不安全的文件残留。本节以 PGP Desktop 9.6 版为例进行介绍。

1. PGP 密钥的创建

整个 PGP 软件的功能非常多,且有些配置起来还比较复杂,在此仅介绍与本章主题有关的文件和信息的加密和数字签名方面的应用。用 PGP 软件进行文件加密和数字签名,实际上就是由 PGP 软件本身为用户颁发包括公、私钥密钥对的证书。所以要使用这款软件首先要做的就是密钥的生成。具体方法如下:

(1) 打开安装好的 PGP 软件,界面如图 9-11 所示。

(2) 选择 File→New PGP Key 菜单命令,或者按 Ctrl+N 组合键,都可打开 PGP Key Generation Wizard(密钥生成向导)界面。按照提示一步步生成密钥后的界面如图 9-12 所示。

2. 公/私钥的获取

根据文件加密和数据签名原理的介绍可以知道,要利用包括公钥和私钥的证书进行文件加密和数据签名,首先就要把自己的公钥向要发送加密邮件的所有接收方发布,让他们知道自己的公钥,否则他们在收到自己的加密邮件时将打不开。整个公钥的发布包括两个主

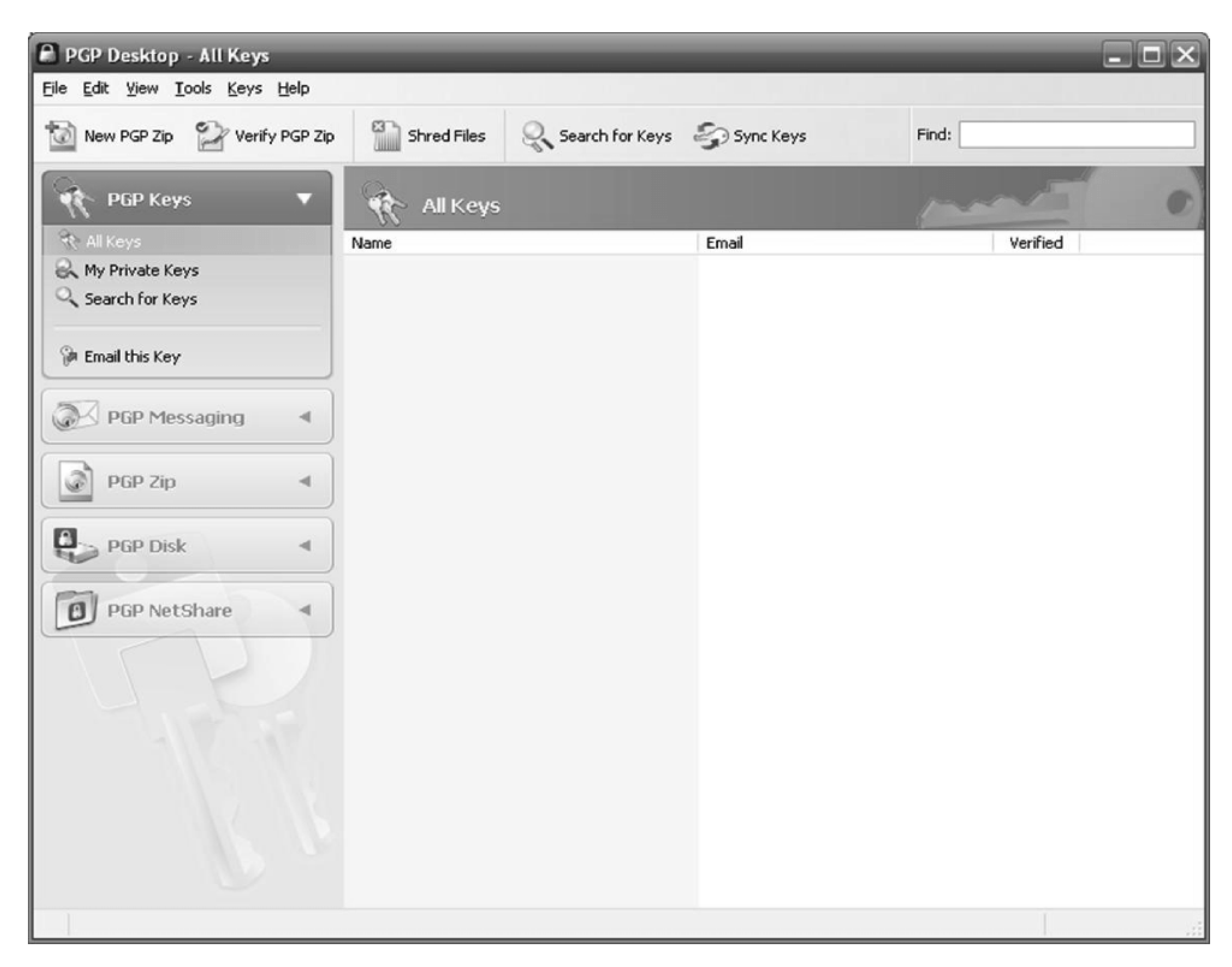


图 9-11　开始界面

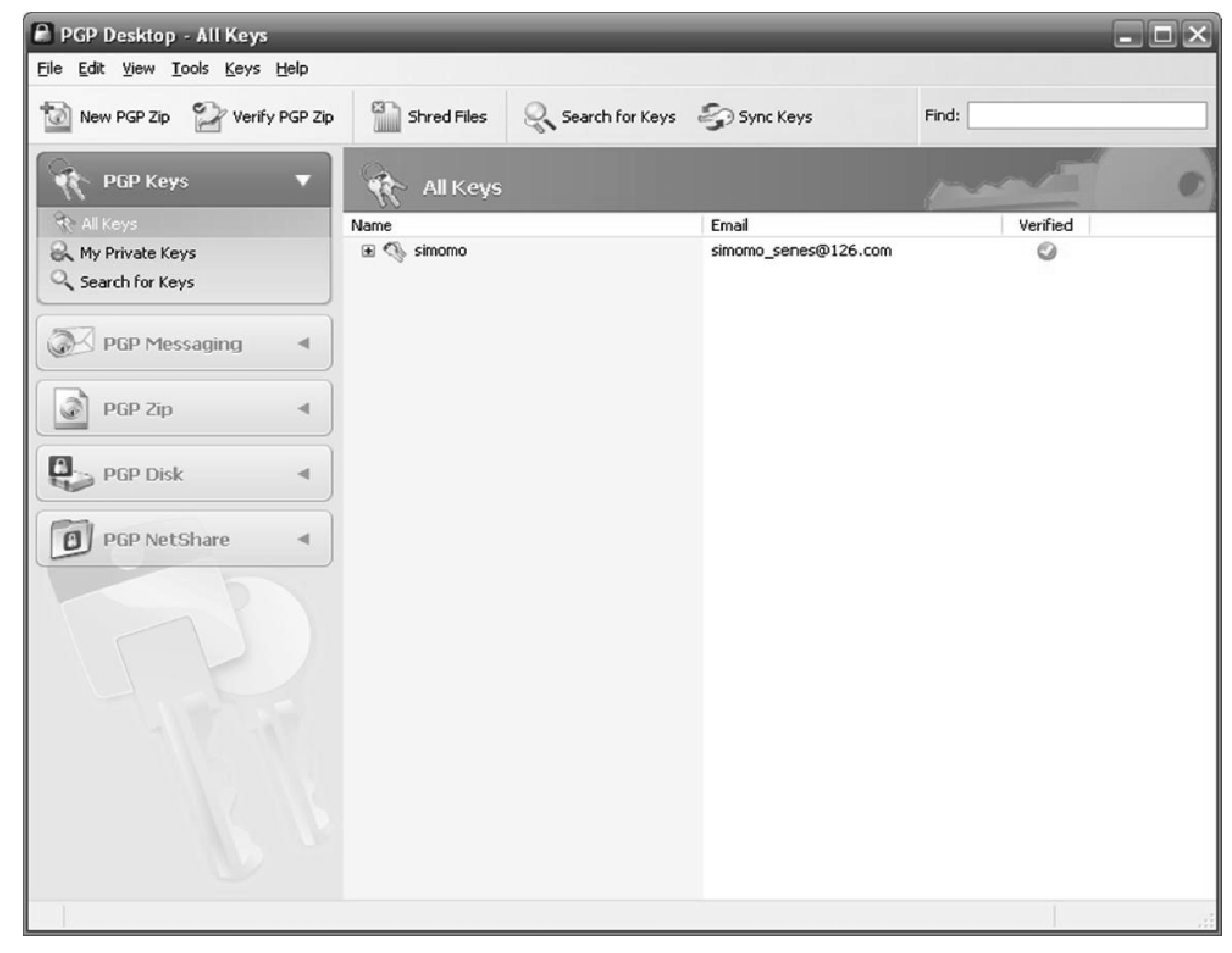


图 9-12　生成密钥界面

要步骤，那就是用户自己先导出公钥文件，发送给好友；然后好友再从接收到的包括公钥文件的附件中导入公钥，用于文件解密。下面分别予以介绍。

1）用户自己导出公钥文件

要发布自己的公钥，首先自己要先从证书中导出。从证书中导出公钥的方法很简单，具体方法如下：

（1）在如图9-12所示窗口中选择自己的一个要用来发送加密邮件的证书（带两钥匙的选项）并右击，在弹出的快捷菜单中选择Export（导出）命令，打开如图9-13所示的Export Key to File（导出公钥到文件）对话框。



图9-13　Export Key to File 对话框

（2）选择保存导出公钥的公钥文件存储位置，然后单击"保存"按钮即可完成公钥的导出。默认的文件格式为.asc。如果选中Include Private Key(s)复选框，则同时导出私钥。因为私钥不能让别人知道，所以在导出用来发送给邮件接收者的公钥中，不要选中此复选框。

公钥导出后就可以通过任何途径（如邮件发送，QQ、MSN点对点文件传输等）向其他用户发送公钥文件，不必担心被人窃取，因为公钥可以被别人知道。

其实还有一种更直接的方法来获取证书的公钥，那就是在如图9-12所示的窗口中，选择对应的证书密钥对，右击并在弹出的快捷菜单中选择Copy Public Key（复制公钥）命令，然后再在任何一个文本编辑器（如记事簿、写字板等）中粘贴所复制的公钥，那可把公钥的真正内容复制下来（如图9-14所示），最后不要做任何修改，以.asc文件格式保存下来。这就是公钥文件。

2）好友导入公钥文件

当好友接收到来自你发送的包括公钥文件的邮件时，他（她）们需要把这个公钥文件导入到自己的计算机上，以便于工作解密时使用，导入步骤如下：

（1）在附件中双击这个公钥文件，打开如图9-15所示的Select key(s)（选择公钥）对话框。在此对话框中显示了公钥文件中包括的公钥。

（2）选中需要导出的公钥（如若有多个的话可以单击Select All按钮全选），单击Import按钮，即可完成公钥的导入。

导入后的好友公钥也会加入到如图9-12所示的All keys（所有密钥）窗口中。要查看

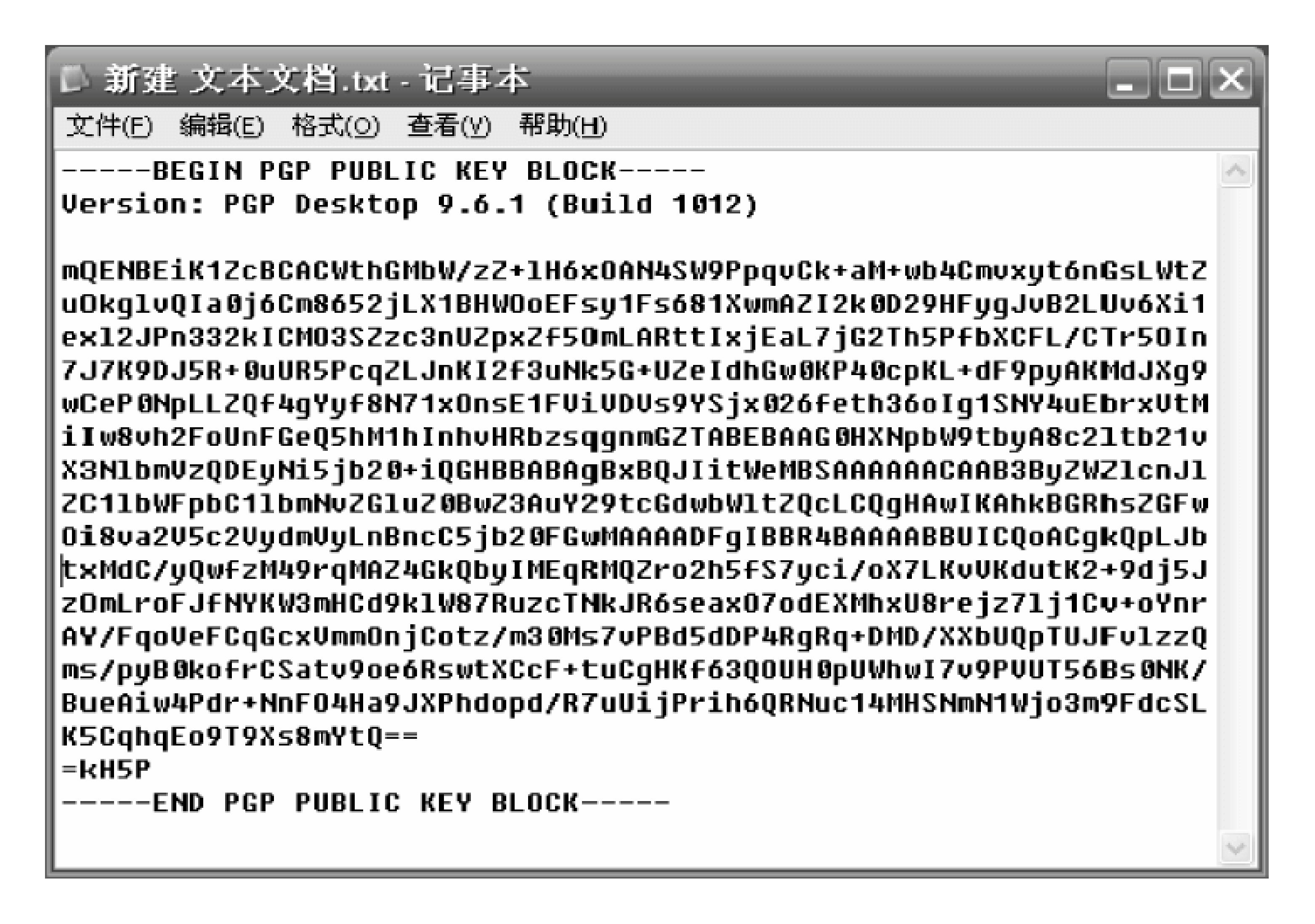


图 9-14　公钥密码

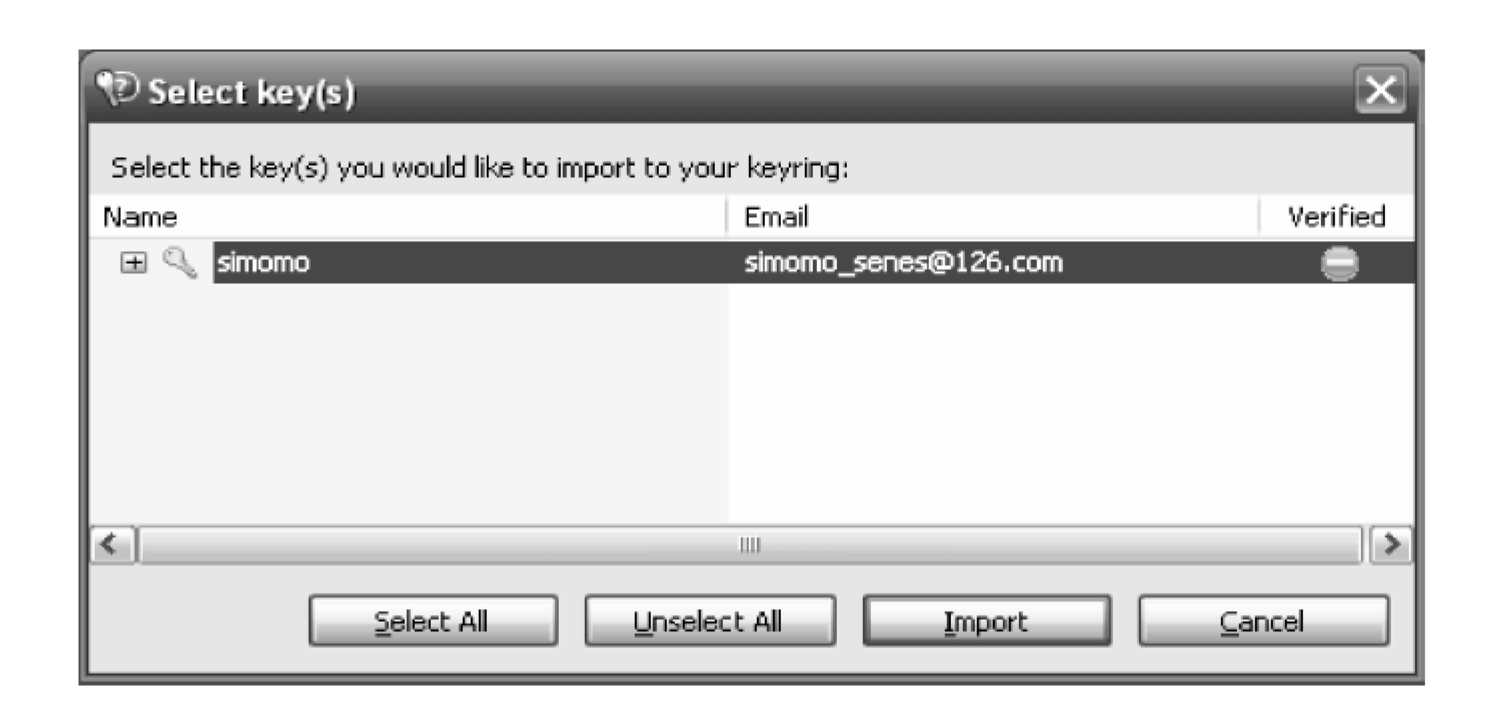


图 9-15　Select key(s)对话框

自己所有的密钥，可选择窗口左边导航栏中的 My Private Keys 选项，在右边详细列表窗格中即可得到。

3. PGP 在文件中的应用

下面将介绍 PGP 软件在文件加密方面的应用。使用 PGP 进行文件加密需要用接收方的公钥进行。具体方法如下：

（1）首先要从对方获取公钥，获取公钥的方法前面已有介绍。可以通过邮件，也可以通过其他传输途径，得到后再导入到自己的 PGP 程序密钥列表中，如图 9-12 所示。

（2）在资源管理器中选择要加密传输的文件（本例为新建文本文档）或文件夹并右击，安装 PGP 程序后在弹出的快捷菜单中都会有一个名为 PGP Desktop 的子菜单，选择其中的 Add “新建文本文档. txt” to new PGP zip 命令（如果要求只签名或者只加密可以选择相应的其他选项），打开如图 9-16 所示的 New PGP Zip 对话框。

（3）单击“下一步”按钮，在 Encrypt 中选择 Recipient keys，选择下一步。双击左框 Key source 中的密钥，使其添加到右框 Keys to add 中，如图 9-17 所示。

单击 OK 按钮，在 Sign and Save 界面中选择 Signing Key 输入密码选择保存地址，继续单击“下一步”按钮，完成即可在相应地址找到加密并签名的文件“新建文本文档. txt. pgp”。

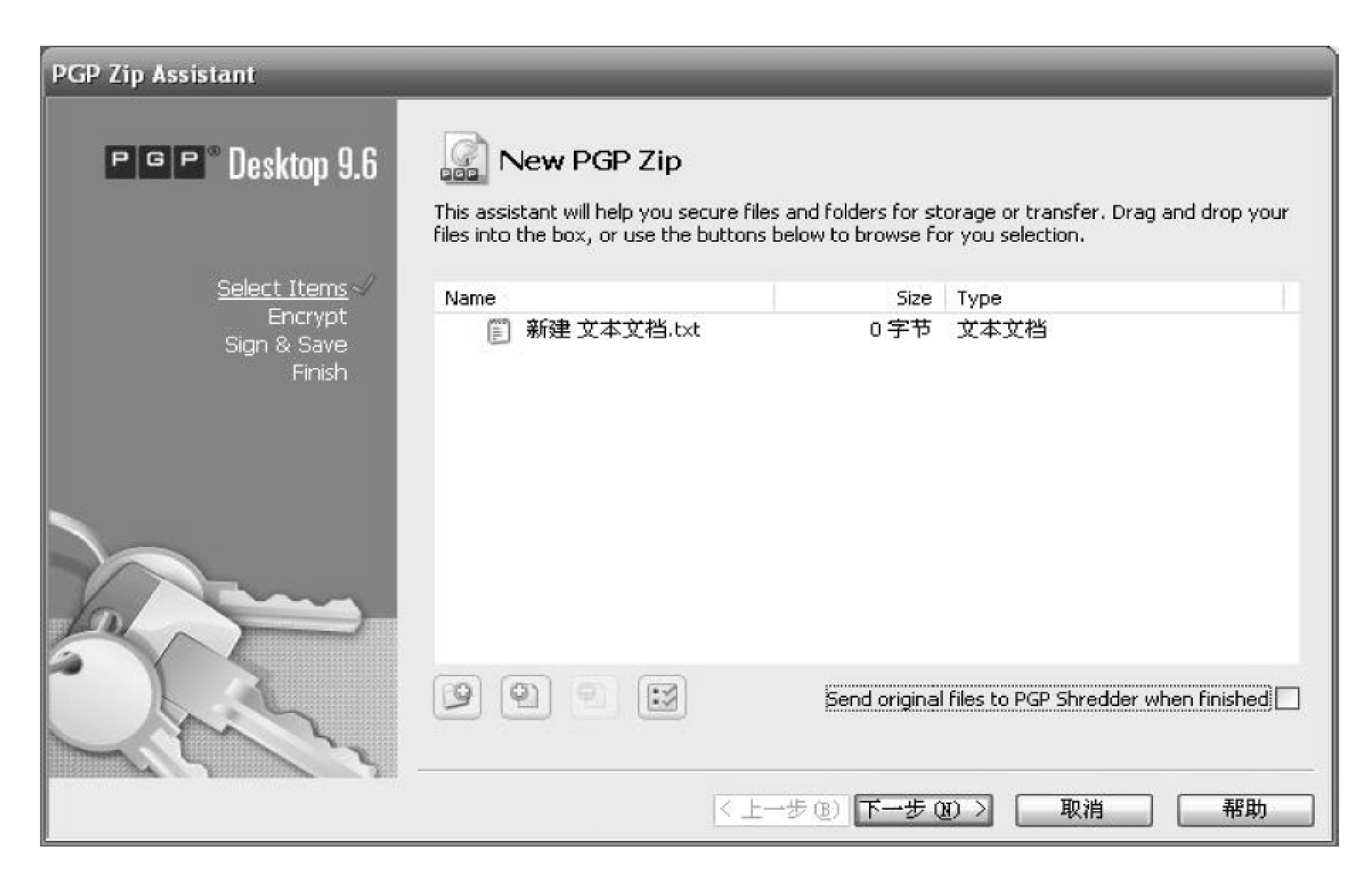


图 9-16　New PGP Zip 对话框

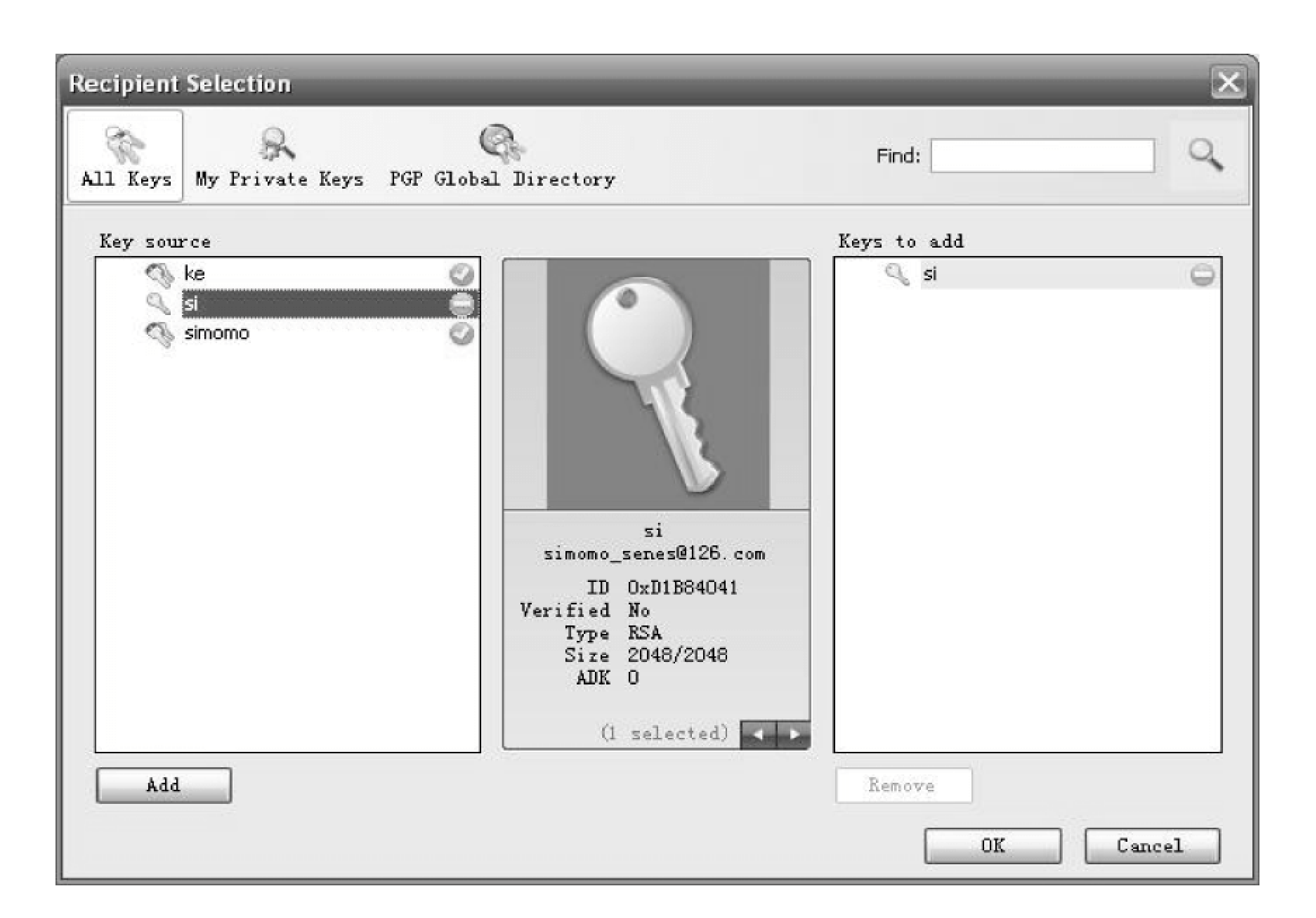


图 9-17　添加密钥

然后把这个加密的文件通过邮件，或者其他方式（如 QQ、MSN 等）传输给接收方。PGP 可以对正在使用的文件动态加密。

当用户收到这封加密了附件文件的邮件后，右击文件，在弹出的快捷菜单中选择 Decrypt&Verify 命令即可得到加密前的文件了。

4. PGP 在邮件中的应用

PGP Desktop Email 在电子邮件发送时自动地加密和签名，并在接收时自动解密。用

户需要做的只是像平常一样发送和接收电子邮件，其余的 PGP Desktop 会做好。PGP Desktop Email 会根据个人或企业设定的安全战略，当用户要发送电子邮件时，会自动地使用接收者的公钥来加密（确保传输过程不会资料泄露），并且使用用户自己的私钥来签名（证明发送者是用户本人）；当用户收到电子邮件时，PGP Desktop Email 会使用用户的私钥将邮件解密，并使用发送者的公钥来验证其签名。如果对方没有安装 PGP Desktop Email，用户原来使用的电子邮件功能不会收到影响。

1）加密邮件的传送

（1）安装完成后，PGP Desktop Email 会自动插入在用户的电子邮件客户端和用户的邮件服务器之间，并观察着电子邮件传输。安装了 PGP Email 的 Product Information 中 PGP Messaging 显示为可用，如图 9-18 所示。



图 9-18　PGP 所安装的组件

（2）当传入邮件到达，PGP Desktop Email 会在他们到达收件箱之前拦截他们，并自动使用用户的私钥解密和使用公钥校验。完成之后 PGP Desktop Email 就会把得到的信息放到收件箱中。大多数情况下，解密后的邮件跟其他邮件并无区别。

（3）当用户发送邮件时，PGP Desktop Email 会在邮件去邮件服务器之前拦截他们，并自动根据用户的配置加密和签名。同样，用户只要用客户端创建并发送自己的邮件，其余一切 PGP Desktop Email 都会处理。

2）详细说明

（1）接收邮件

PGP Desktop Email 基于邮件所包含的内容对它进行处理。

没有进行加密和签名：PGP Desktop Email 什么都不做，直接传送。

加密 & 不签名：PGP Desktop Email 会先查找能解密这封邮件的私钥，找到后用其解密并把解密后的邮件放到邮件客户端；如果找不到就会传送仍然加密的邮件。

签名 & 不加密：PGP Desktop Email 会先查找能校验这个签名的公钥，找到合适的公钥后，用其检验，并把解密后的邮件放到邮件客户端；如果找不到就会传送仍然签名的邮件。

签名并加密：PGP Desktop Email 会先找私钥解密消息，再找公钥验证它。

邮件接收后 PGP Desktop Email 会有如下提示信息（如图 9-19 所示）。

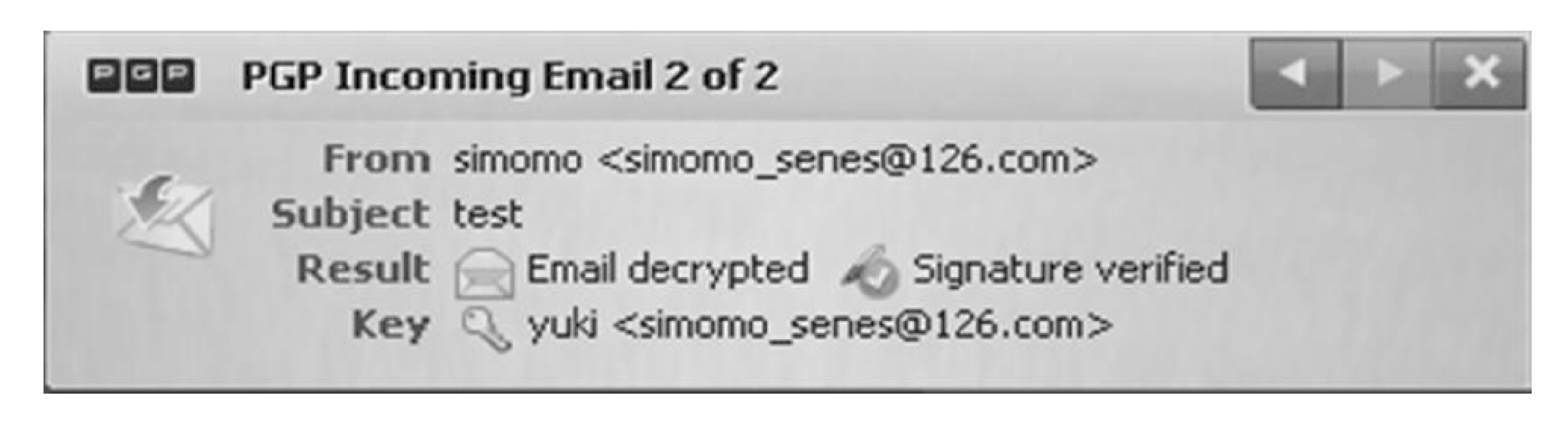


图 9-19　PGP 邮件接收信息

（2）发送邮件

PGP Desktop Email 包括以下四个默认的设置。

Mailing List Admin Requests：要求发送邮件不加密和签名。

Mail List Submissions：要求发送邮件签名不加密。

Require Encryption：[PGP] Confidential：所有在邮件客户端标记机密或者在主题中包含"[PGP]"的邮件，必须用一个有效的公钥加密，否则就不发送它。这个设置使得发送加密邮件容易控制。

Opportunistic Encryption：说明了如果一个消息的密钥找不到，则不加密发送。这个设置保证了即使给收件人的加密密钥无法找到，邮件也会被发送（除非信息标志为机密）。

具体操作为：在 PGP Messaging 界面中选择相应的 Service，然后在 Security Policies 一栏中选择 Edit Policies 即可选择相应的设置。当发送的信息满足其中某个设置的条件时，邮件就会按照该设置发送邮件。

邮件发送后 PGP Desktop Email 会有如下提示信息（如图 9-20 所示）。



图 9-20　PGP 邮件发送信息

9.5　S/MIME

安全多功能互联网协议邮件扩展协议（Secure Multipurpose Internet Mail Extensions，S/MIME）是从 PEM 和 MIME 发展而来的。它集成了三类标准：MIME（RFC 1521）、加密消息语法标准（Cryptographic Message Syntax Standard，CMSS）和证书请求语法标准（Certification Request Syntax Standard，CRSS）。

9.5.1　MIME 对 RFC 822 的扩充

早期的 Internet 电子邮件的格式是由 RFC 822 定义的。RFC 822 文档定义了邮件内容的主体结构和各种邮件头字段的详细细节，但是，它没有定义邮件体的格式，RFC 822 文

档定义的邮件体部分通常都只能用于表述一段普通的文本，只能是 ASCII 字符，而无法表达出图片、声音等二进制数据。

由于 Internet 的迅猛发展，人们已不满足于电子邮件仅仅是用来交换文本信息，而希望使用电子邮件来交换更为丰富多彩的多媒体信息，例如，在邮件中嵌入图片、声音、动画和附件。但是图片和声音等内容是非 ASCII 码的二进制数据，而 RFC 822 邮件格式只适合用来表达纯文本的邮件内容，所以，要使用 RFC 822 邮件格式发送这些非 ASCII 码的二进制数据时，必须先采用某种编码方式将它们“编码”成可打印的 ASCII 字符后再作为 RFC 822 邮件格式的内容。邮件阅读程序在读取到这种经过编码处理的邮件后，再按照相应的解码方式解码出原始的二进制数据，这样就可以借助 RFC 822 邮件格式来传递多媒体数据了。这种做法需要解决以下两个技术问题：①邮件阅读程序如何知道邮件中嵌入的原始二进制数据所采用的编码方式；②邮件阅读程序如何知道每个嵌入的图像或其他资源在整个邮件内容中的起止位置。MIME 就是针对这种情况出现的。

RFC 822 定义了邮件由若干个 Header field 和一个 Body 两部分组成，IETF 在 RFC 2045～RFC 2049 中定义的 MIME 标准，补充定义了几个 Header field，并对 Body 定义了复杂的格式，邮件体除了 ASCII 字符类型之外，还可以包含其他多种数据类型，从而使得电子邮件可以携带更多更复杂的信息，用户可以使用 MIME 增加非文本对象，比如把图像、音频、格式化的文本或微软的 Word 文件加到邮件体中去。MIME 中的数据类型一般是复合型的，也称为复合数据。由于允许复合数据，用户可以把不同类型的数据嵌入到同一个邮件体中。在含有复合数据的邮件体中，设有边界标志，它标明每种类型数据的开始和结束。MIME 对 RFC 822 是向下兼容的。MIME 和 RFC 822 的关系如图 9-21 所示。

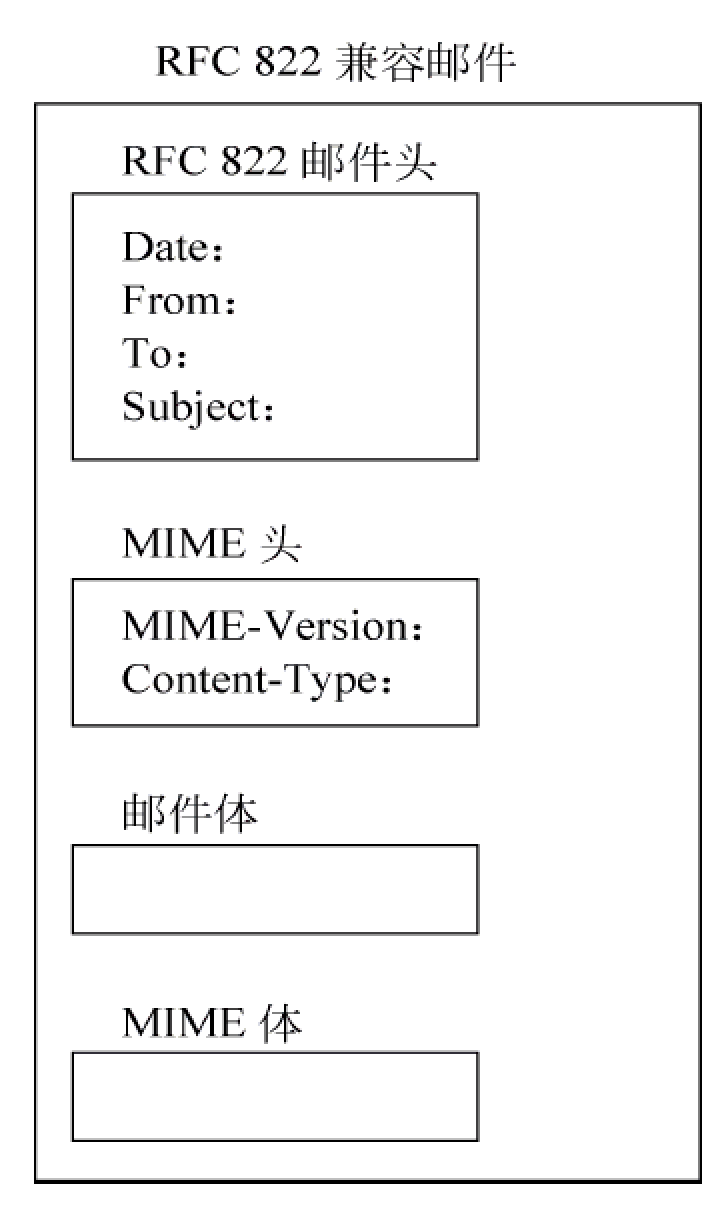


图 9-21　邮件头部字段

MIME 定义了 5 个新的 Header field，除了 MIME-Version 以外，其他 4 个以 Content 开头，用于描述邮件 Body 部分所包含的信息（见表 9-4）。

① MIME-Version（MIME 的版本）：现在版本为 1.0。若无此行则为英文文本。

② Content-Type（内容类型）：说明邮件的性质。

③ Content-Transfer-Encoding（内容传送编码）：说明在传输时邮件的主体如何编码。

④ Content-ID（内容 ID，可选）：邮件的唯一标识符。

⑤ Content-Description（内容描述，可选）：对于消息主体中对象的正文描述，在该对象不可读的情况下，这个字段非常有用（如音频数据）。

MIME 标准规定 Content-Type 说明含有两个标识符，即内容类型（type）和子类型（sub-type）。其中，基本内容类型有 7 种，子类型有 15 种。因此，MIME 格式的邮件中可以包含多种信息，当 MIME 类型为复合类型时，还可以实现数据的并行、嵌套，从而可以表示更加复杂的信息。

表 9-4 MIME 内容的类型

类　　型	子　类　型	描　　述
Text	Plain	无格式的正文
	Enriched	提供了更大的格式灵活性
Multipart	Mixed	按规定顺序的几个独立部分
	Parrallel	必须同时读取的几个部分
	Alternative	不同格式的同一邮件
	Digest	第一部分是一个完整的 RFC 822 邮件
Message	Rfc822	MIME RFC 822 邮件
	Partial	为传输将大的邮件分开
	Extended-body	邮件必须从其他位置获得
Image	Jpeg	JFIF 编码,JPEG 格式的静止图像
	Gif	GIF 格式的静止图像
Video	Mpeg	MPEG 格式的影片
Audio	Basic	声音
Application	PostScript	Adobe Postscript 可打印文档
	Octet-stream	通常的 8b 字节组成的二进制数据

对于消息主体的 text type(正文类型),除了需要对指定字符集的支持外,不需要专门的软件来获得正文的含义。主要的子类型是 plain text(纯文本),就是简单的由 ASCII 字符或 ISO 8859 字符组成的字符串。Enriched(丰富的)子类型允许更大的格式灵活性。

Multipart type(多部分类型)指示消息主体包含了多个独立的部分。Content-type 首部字段包含了称为 boundary(边界)的参数,定义了在消息的不同部分之间的分隔符,这个边界不应该出现在消息的任何一部分内容之中。每个边界开始于一个新行,由两个字符跟着一个分隔符组成。指示着最后一个部分结束的最后一个边界还有两个连字符作为后缀。在消息主体的每个部分,可以存在可选的普通的 MIME 首部。

Message type 提供了 MIME 的一个重要功能。Message/rfc 822 subtype 指示消息主体是完整的消息,包括首部和主体。尽管使用了这种子类型名,但包装的消息可以不仅仅是简单的 RFC 822 消息,而是任何 MIME 消息。

Application type 指的是其他种类的数据,典型的是邮件应用程序不理解的二进制数据或信息。

Content-Transfer-Encoding 标识嵌在邮件中的二进制数据是怎样被编码的,目前有 7 种不同的编码方式:7bit、8bit、binary、quoted-printable、base64、ietf-token 和 x-token。最常用的对二进制编码方式是 base64 方式,该方式中每 6 位二进制数据被映射为一个 8 位长的 ASCII 文本字符。

9.5.2 S/MIME 对 MIME 类型的扩充

MIME 允许对其 Content-Type 进行扩充,S/MIME 就是在其基础上增加了几种新的 MIME 子类型:S/MIME 对安全方面的功能进行了扩展,它可以把 MIME 实体(比如数字签名和加密信息等)封装成安全对象。S/MIME 增加了几种新的 MIME 子类型,用于提供数据保密、完整性保护、认证和鉴定服务等功能(见表 9-5)。

表 9-5　S/MIME 内容类型

类 型	子 类 型	S/MIME 参数	描 述
Multipart	Signed	—	清除签名的消息分成两个部分：一个是消息部分，另一个是签名部分
Application	pkcs7-mime	signedData	签名的 S/MIME 实体
	pkcs7-mime	envelopedData	加密的 S/MIME 实体
	pkcs7-mime	degenerateSignedData	只包含了公开密钥证书的实体
	pkcs7-signature	—	多部分/签名消息的签名子部分的内容类型
	pkcs10-mime	—	证书注册请求消息

9.5.3　S/MIME 中密码算法的应用

表 9-6 总结了 S/MIME 中使用的加密算法。S/MIME 使用了下面两个取自 RFC 2119 的术语来说明级别的要求。

MUST(必须)：这个定义是协议的必须要求。

SHOULD(应该)：建议实现包括这个特征和功能。

表 9-6　S/MIME 中使用的加密算法

功　　能	需　　求
创建用于形成数字签名的消息摘要算法	必须支持 SHA-1 和 MD5 应该使用 SHA-1
加密消息摘要以形成数字签名	发送和接收代理必须支持 DSS 发送和接收代理应该支持 RSA 加密 接收代理应该支持使用长度为 512 到 1024 比特的密钥来验证 RSA 的签名
加密会话密钥和消息一起传送	发送和接收代理必须支持 Diffie-Hellman 发送代理应该支持使用长度为 512 到 1024 比特的 RSA 加密 接收代理应该支持 RSA 解密
使用一次性会话密钥来加密传输的消息	发送代理应该支持三重 DES 和 RC2/40 的加密 接收代理应该支持三重 DES 的解密，必须支持 RC2/40 的解密

只需简单地看一下密钥的长度就能知道哪种算法更好：密钥越长，安全性就越高。然而接收发双方并不总是处在同一水平线上。例如，发送方代理可能试图使用 RC2/128 来加密从而增加安全性；然而接收方可能只具有 RC2/40 解密的能力。因此 S/MIME 协议定义了一个过程，当要发送 S/MIME 消息时，该过程可以定义一个最好的算法。下面是发送方代理做决策时应该使用的一些指定的规则：

(1) 已知能力。如果发送代理在此前接收到了接收方一个密码学功能的列表，则发送方应该选择列出的第一个功能来加密要发送的数据。

(2) 未知能力但已知使用了加密。如果发送方代理对接收方代理的解密能力不清楚，但至少从接收方接收过一条以前曾经加了密的消息，则此时发送代理应该使用以前的那种

算法来加密要发送的消息。

(3) 未知能力且未知 S/MIME 版本。当发送方以前没有与接收方联系过,也不知接收方的能力时,如果发送方愿意冒着接收者可能不能解密消息的危险,则应该使用三重 DES;如果不愿冒这个险,那么发送方必须使用 RC2/40。

9.5.4 S/MIME 消息

S/MIME 消息准备的过程是首先安全化一个 MIME 实体,然后按照 S/MIME 内容类型来包装数据。

S/MIME 使用签名、加密或同时使用两者来保证 MIME 实体的安全。一个 MIME 实体可能是一个完整的消息(除了 RFC 822 首部),或者如果 MIME 内容类型是多部分的,那么一个 MIME 实体是消息的一个或多个子部分。MIME 实体按照 MIME 消息准备的一般规则来准备。然后,该 MIME 实体加上一些与安全有关的数据,如算法标识符和证书,被 S/MIME 处理以生成 pkcs 的对象。然后 pkcs 对象被看做消息内容并包装成 MIME。

S/MIME 内容类型有封装数据(enveloped data)、签名数据(signed data)、清除签名(clear signing)和封装且签名的数据(enveloped-and-signed data)。对不同的数据类型其包装过程也不一样。

准备一个封装数据的 MIME 实体步骤如下:

(1) 特定的对称加密算法(RC2/40 或 3DES 算法)生成伪随机的会话密钥;

(2) 对每个接收者,使用接收者的公开 RSA 密钥对会话密钥进行加密;

(3) 对每个接收者准备称为 RecipientInfo(接收者信息)的数据块,该块中包含了发送者的公开密钥证书、用来加密会话密钥算法的标识以及加密的会话密钥;

(4) 使用会话密钥加密消息的内容。

RecipientInfo 后面跟着加密的内容组成了包装数据。然后使用 radix-64 对这个信息编码。简单消息(去掉 RFC822 的首部)如下所示:

```
Content - Type: application/x - pkcs7 - mime; smime - type = enveloped - data;
boundary = " ---- = _NextPart_000_0027_01C8EBD2.D498C3A0";
name = "smime.p7m" Content - Transfer - Encoding: base64
Content - Disposition: attachment; filename = "smime.p7m"
DAXyJubDNnvRcFUIQY5l + zKSXy2C1OaxMdgXKkupTUAEkEbk41XZhoq5IZtGIki9Eb/Sza181dXTVlPfaFIjgXB/
Qx + VnQr + ncUvrCeic2lQsHCSK43LCg/Ic1dv4vtrV5S/1U3zDxni2VQtjKuksZIKjifcPvFNDzJJSf7icp0pqNu
Rw7Aggive3 + mOhkwMF5amUEdV8uLSBDi/cUYxj + ExN8V/Ul9Yatco + meGoYx7sq + q3enaPuTE913bcCBFKteY + +
nuMs6//qHCYkVMCYcznwAAAAAAAAAAAA =
```

为了恢复加密的消息,接收者首先去掉 base64 编码,然后使用接受者的私有密钥来恢复会话密钥。最后使用会话密钥来解密消息内容。

对于签名数据,准备一个 MIME 实体的过程如下:

(1) 选择签名算法(SHA-1 或 MD5);

(2) 计算需要签名内容的消息摘要;

(3) 使用发送者的私有密钥加密消息摘要;

(4) 准备成为 SignerInfo(签名者信息)的数据块,该块中包含了签名者的公开密钥证书、消息摘要算法的一个标识符、用来加密消息摘要算法的标识符以及加密的消息摘要。签

名数据实体包括消息摘要算法标识符、被签名的消息和 SignerInfo。然后使用 base64 进行编码。简单消息(去掉 RFC 822 的首部)如下所示：

```
Content-Type: application/x-pkcs7-mime;smime-type=signed-data;
boundary="----=_NextPart_000_003A_01C8EB4D.5C942140";name="smime.p7m"
Content-Transfer-Encoding: base64
Content-Disposition: attachment;filename="smime.p7m" DA3MjEwODE4MDdaMCMGCSqGSIb3DQEJBDE-
WBBQXJTyFVhGaGMDB/153dudhFIG4zDBbBgkqhkiG9w0BCQ8xTjBMMAoGCCqGSIb3DQMHMA4GCCqGSIb3DQMCAgIAg-
DANBggqhkiG9w0DAgIBQDAHBgUrDgMCBzANBggqhkiG9w0DAgIBKDAHBgUrDgMCHTANBgkqhkiG9w0BAQEFAARAtct-
V82taQdomI/xuwBQ4AsM6pRu8MV7X2KM9DA3JNRuh+hbtv6DQm9hM/BZb9fs27ZWao//NxsTq0Sr1uHBK+wAAAA-
AAAA==
```

为了恢复签名的消息和验证签名，接收者首先要除掉 base64 编码，然后使用签名者的公开密钥来解密消息摘要。接收者单独计算消息的摘要并且将它与解密后的消息摘要相比较来验证签名。

发送方已经签名的数据可能会被一个与 S/MIME 不兼容的接收者收到，这样会导致初始的内容不可用。为解决这个问题，S/MIME 使用一个可供选择的结构 multipart/signed 类型。

Multipart/signed 类型的主体由两个部分组成。第一个部分可以是任意的 MIME 内容类型，以明文的形式保留并置于最后的消息中。第二部分的内容是签名数据的一种特殊情况，称为独立签名，它省略了可能包含在签名数据里面的明文的复制。简单消息(去掉 RFC 822 的首部)如下所示：

```
Content-Type: multipart/signed; protocol="application/x-pkcs7-signature";
micalg=SHA1; boundary="----=_NextPart_000_0064_01C8EB4D.C55573A0"
------=_NextPart_000_0064_01C8EB4D.C55573A0
Content-Type: application/x-pkcs7-signature;
name="smime.p7s" Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename="smime.p7s"
MIAGCSqGSIb3DQEHAqDAKBggqhkiG9w0DBzAOBggqhkiG9w0DAgICAIAwDQYIKoZIhvcNAwICAUAwBwYFKw4D-
AgcwDQYIKoZIhvcNAwICASgwBwYFKw4DAh0wDQYJKoZIhvcNAQEBBQAEQGautHyg0UAUBBCCy9CX6xubc5URywX +
tgFPyE/XTMt200024qPSaaBVcc2tCp1jwTO9OAugYDRWJ99FaqQ0acUAAAAAAAA=
------=_NextPart_000_0064_01C8EB4D.C55573A0--
```

这时候准备 MIME 实体的过程可以先加密数据后签名，也可以先签名后加密数据。即嵌套使用 envelopedData 和 SignedData。简单消息(去掉 RFC 822 的首部)如下所示：

```
Content-Type: application/x-pkcs7-mime;
smime-type=enveloped-data;
boundary="----=_NextPart_000_0035_01C8EBD4.69ACDE30";
name="smime.p7m"
Content-Transfer-Encoding: base64
Content-Disposition: attachment;
filename="smime.p7m"
LnT1jIez6znCAujjbkJNRe1uVf6ltkuFQ7PnICMs3IXWNFF73gv+O+TqRoCSHGTi/Y6z0HV9ZWDoeVXoTrhU98-
U7zQr4YweRibdVNDtMGGm+pgPLd81nqpW8iZRRDly6nJKuvM8lusWw/AjZfGtrVKA0yKtbnzH+hzRik2YaE2Npc-
ijDsstgr9+NZlWWo5xVXNN/lWxo3YHahj0528Gt/WsKRADpuSJrPdr0ajjP5id5IWJ54hyx8oVqXC7EBhdTVAZj-
OeAD9DvmdLOZB2YirJi7eRhcb/62uAAAAAAAAAAAAAA==
```

9.5.5 S/MIME 证书的处理

S/MIME 使用符合 X.509 版本 3 标准的公开密钥证书。S/MIME 使用的密钥管理方法是严格的 X.509 证明层次和 PGP 的信任网络的混合。S/MIME 的管理者必须为用户配置可信任的密钥表和废除证书的列表。证书是被认证机构签了名的。

S/MIME 用户可以完成以下密钥管理功能。

(1) 密钥的生成：与管理有关的使用程序的用户必须能够生成单独的 Diffie-Hellman 和 DSS 密钥对，并且应该能够生成 RSA 密钥对。每个密钥对必须从一个好的不确定的随机输入源生成，并且采用安全的方式进行保护。用户代理应该生成 768～1024bit 间的密钥对，并且一定不能生成小于 512b 的密钥对。

(2) 注册：用户的公开密钥和认证一起注册获得 X.509 公开密钥证书。

(3) 证书的存储和查询：用户需要访问证书的本地列表来验证进入的签名和输出的消息加密。这样的一张表可以让用户进行维护。

X.509 是一个重要的标准，基于公开密钥加密和数字签名，这个标准没有专门指定使用的加密算法，但推荐使用 RSA。其核心部分是与每个用户联系的公开密钥证书，实际上 X.509 证书是一种由发布者数字签名的用于绑定某种公开密钥和其持有者身份的数据结构。证书、数字证书、电子证书等都是 X.509 证书的同义词，有时也称为公钥证书，是一种权威性的电子文档，尤其具有权威性、可信任性及公正性的第三方机构(CA)颁发。数字证书包括以下要素。

- 版本：默认的版本是第 1 版。
- 序列号：在发布证书中的一个有唯一性的整数值，是与这张证书有明确联系的。
- 颁布者名字：创建和签名这个证书的第三方机构(CA)的 X.509 名字。
- 有效期：由两个日期组成，证书的有效起始日期和结束日期。
- 主体名：这张证书提及的用户名。
- 主体的公开密钥信息：主体的公开密钥加上这个密钥使用算法的标识符，以及算法的相关参数。
- 签名：它保护用 CA 私有密钥加密的其他字段的散列码。

9.5.6 增强的安全服务

目前，可以使用三种可选的增强的安全服务来扩展当前的 S/MIMEv3 安全以及证书处理服务。

签名收据：签名收据是一种可选的服务，它考虑的是消息发送的证明。收据为发送者提供了一种向第三方出示证明的手段：接收者不仅收到了消息，而且验证了初始消息的数字签名。最后，接收者对整个消息以及相应的签名进行签名作为接收的证明。该服务仅仅用于签名的数据。

安全标签：安全标签可以通过两种方式来使用。第一种，也可能是最容易识别的方法，就是描述数据的敏感级。这可以使用一个分级的标签列表(机密、秘密、限制等)。另一项技术是使用标签来控制授权和访问，描述哪一类接收者可以访问数据。

安全邮件列表：当 S/MIME 协议提供它的服务时，发送代理必须为每一个接收者创建特定于接收者的数据结构。随着某一个特定消息的接收者的数目的增加，这一处理可能会削弱发送消息的性能。这样，邮件列表代理可以接收一个单独的消息并针对每一个接收者完成特定于接收者的加密。

9.5.7 S/MIME 的应用

数字证书也叫数字标识(Digital ID)，一般由权威公正的第三方机构即 CA 签发，数字证书将用户身份、公钥通过数字签名绑定。公钥所对应的私钥由用户秘密持有，公钥和私钥构成了公钥加密算法的密钥对，可用来加解密或进行数字签名。

数字证书有代码签名证书、安全电子邮件证书、个人和单位身份证书以及服务器证书等几种类型，分别应用于不同的场合。例如代码签名证书主要用于给程序签名；安全电子邮件证书用于给邮件数字签名；而个人数字证书用途则很广，可以用来给 Office XP 文档、软件代码、XML 文件、Email 等文件签名。数字证书可以通过网络向相关 CA 机构申请获得，或者使用 makecert 之类的工具自己生成。数字证书可以存放在计算机的硬盘、软盘或 IC 卡中。

1. 获取数字证书

获得数字 ID 有多种方法。有供教学模拟的数字证书生成软件，也可以在线申请。另外，Windows 2003 Server 具备证书组件。

如果用命令行工具 makecert 生成数字证书，应首先下载微软的 Authenticode for Internet Explorer 5.0，该工具包中有个 makecert.exe 程序，可以用它来制作自己的数字证书。这种方法生成的数字证书虽然不属于受信任的证书，但能够正常使用，操作步骤是：在命令提示行中，用 CD 命令进入 makecert.exe 所在目录，然后输入以下命令：

```
makecert [options] outputCertificateFile
```

Makecert.exe 包含基本选项和扩展选项。基本选项是最常用于创建证书的选项；扩展选项提供更多的灵活性。主要的基本选项包括：

- -n x509name 指定主题的证书名称。此名称必须符合 X.509 标准。最简单的方法是在双引号中指定此名称，并加上前缀 CN =；例如，"CN=myName"。
- -pe 将所生成的私钥标记为可导出。这样可将私钥包括在证书中。
- -sk keyname 指定主题的密钥容器位置，该位置包含私钥。如果密钥容器不存在，系统将创建一个。
- -sr location 指定主题的证书存储位置。Location 可以是 currentuser(默认值)或 localmachine。
- -ss store 指定主题的证书存储名称，输出证书即存储在那里。
- -# number 指定一个介于 1 和 2,147,483,647 之间的序列号。默认值是由 Makecert.exe 生成的唯一值。
- -$ authority 指定证书的签名权限，必须设置为 commercial(对于商业软件发行者使用的证书)或 individual(对于个人软件发行者使用的证书)。

主要的扩展选项包括：

- -a algorithm 指定签名算法。必须是 md5(默认值)或 sha1。
- -b mm/dd/yyyy 指定有效期的开始时间。默认为证书的创建日期。
- -cy certType 指定证书类型。有效值是 end(对于最终实体)和 authority(对于证书颁发机构)。
- -d name 显示主题的名称。
- -e mm/dd/yyyy 指定有效期的结束时间。默认为 12/31/2039 11：59：59 GMT。
- -ir location 指定颁发者的证书存储位置。Location 可以是 currentuser(默认值)或 localmachine。
- -is store 指定颁发者的证书存储名称。
- -sv pvkFile 指定主题的.pvk 私钥文件。如果该文件不存在，系统将创建一个。

例如：makecert testCert.ce 创建了一个由默认测试根颁发的测试证书，并将其写入 testCert.cer。makecert -ss testCertStore 创建了一个由默认测试根颁发的证书并将其保存到证书存储区。makecert -sv testCert.pvk -ss testCertStore testCert.cer 创建了一个由默认测试根颁发的证书和一个.pvk 文件，并将此证书同时输出到存储区和该文件。makecert-sv lacl.pvk-n "CN=lacl" -ss My-b 01/01/2005-e 01/01/2095 lacl.cer 执行后，将弹出一个窗口，要求设置私匙密码，如果不设密码，单击“无”按钮即可。于是在 makecert.exe 程序所在目录下，就生成了数字证书文件 lacl.cer 和对应的私匙文件 lacl.pvk。

可以在网上向一些 CA 申请数字证书，例如中国数字认证网(http：//www.ca365.com)、广东省电子商务认证中心(http：//www.cnca.net/)、博大证书(http：//ca.foxmail.com.cn/)、天威诚信(http：//www.itrus.com.cn)等。这里用网证通 NETCA 电子认证系统作为例子 https：//testca.netca.net/ 。登录后，单击“证书申请”，选择“试用型个人数字证书申请”，注意只有安装了根证书(证书链)的计算机，才能完成后面的申请步骤和正常使用申请的数字证书。按照提示，选择“安装试用 CA 证书链”。安装成功出现提示框后，可以看到一个表单。按照表单上的提示，输入完整的个人资料。选择加密服务提供程序(Cryptographic Service Provider，CSP)，这里，CSP 负责创建密钥、吊销密钥，以及使用密钥执行各种加密、解密操作。每个 CSP 都提供了不同的实现方式。当使用特别的数字证书存储介质(如：智能 IC 卡或 USB 电子令牌)存储数字证书及其相应的私有密钥时，可以在“加密服务提供程序(CSP)”下拉框中选择该存储介质生产厂商提供的 CSP。这里可以选择“Microsoft Base Cryptagraphic Provider V1.0”。补充信息可以选填：有效证件类型、证件号码、出生日期、性别、住址、通信地址、通信所在地邮政编码、联系电话、传真号码、存储介质等。然后单击按钮提交。进行上述步骤后，系统将发一封申请成功的信件到申请时使用的邮箱内，其中包括业务受理号和密码，数字证书下载的地址。点击数字证书下载地址，填写业务受理号和密码进行下载。

如果要在已经安装了 Windows 的服务器上安装证书服务以创建证书颁发机构，需要打开“控制面板”中的“添加或删除程序”。选择了安装证书服务之后，“证书服务安装向导”将指导完成安装过程。默认情况下，证书授权机构为用户和管理员提供了可用的网页用于执行与申请证书相关的多种任务，这些网页位于 http：//servername/certsrv，其中 servername 是驻留证书颁发机构的服务器名称。

2. 查看、导入、导出、删除数字证书

用 MMC 来查看。选择“开始”→“运行”命令，输入 MMC 打开“控制台”，选择“文件”→“添加/删除管理单元”命令，在弹出的对话框中单击“添加”按钮，然后在管理单元列表中双击“证书”，选中“我的用户账户”，再单击“完成”按钮，回到“控制台根节点”，窗口下面显示就是当前用户的所有证书。

用 certmgr. msc 查看。选择“开始”→“运行”命令，输入 certmgr. msc，在打开的窗口中，单击“个人”下面的“证书”，可看到所有证书。

用 Certmgr 查看。Authenticode for Internet Explorer 5.0 工具包里面有个 Certmgr. exe 程序，可以查看计算机中的数字证书。双击 Certmgr. exe，就会弹出一个窗口，显示数字证书列表。

在 Internet 选项中查看。运行 Internet Explorer，选择“工具”→“Internet 选项”命令，打开“内容”选项卡，单击“证书”按钮，也可以查看本机上的数字证书。在“证书”对话框中，打开“个人”选项卡，即可看到个人数字证书列表；选定某个数字证书，单击“查看”按钮，可以看到该数字证书的详细信息。

在证书窗口，也可以导出数字证书、导入数字证书、删除数字证书。

3. 保护电子邮件

数字证书最常见的应用就是发送安全电子邮件，即利用安全电子邮件数字证书对电子邮件签名和加密。安全电子邮件证书中包含证书持有者的电子邮件地址、公钥及 CA 中心的签名。使用安全电子邮件证书可以收发加密和数字签名邮件，保证电子邮件传输中的机密性、完整性和不可否认性，确保电子邮件通信各方身份的真实性。证书可以存储在硬盘、USB 中。需要注意的是，证书中的邮件地址必须同绑定的邮件账号一致。

在 Outlook Express 中设定邮件账户。打开 Outlook Express，然后选择“工具”→“账户”命令，出现“Internet 账户”对话框，单击右边的“添加”按钮，选择“邮件”选项。输入邮件显示姓名，单击“下一步”按钮，输入电子邮件地址；单击“下一步”按钮，分别输入接收邮件服务器和发送邮件服务器的域名或 IP 地址。继续单击“下一步”按钮，系统让输入登录到邮箱的账号和密码，建议选中“使用安全密码验证”复选框。继续单击“下一步”按钮，可以看到成功设置了一个新的账户。

邮箱与数字证书的绑定。在 Outlook Express 6.0 中选择“工具”→“账号”命令，选取“邮件” 选项中的邮件账号，即刚才建立的账号，然后单击“属性”按钮。选择上面的“安全”标签，可以看到“签署证书”和“加密首选项”两栏。在“签署证书”项后，单击“选择”按钮。选择数字证书，单击“确定”按钮完成邮箱与证书的绑定。如果单击“选择”按钮，没有相关的证书弹出来，请确认证书已经正确安装且没有过期，同时要确认在 Outlook Express 中所设置的邮箱与在申请数字证书时所提供的邮箱一致。按照同样的方法，也可以在“加密首选项”中把自己的证书选中。最后设置安全邮件高级选项，选择“工具”→“选项”→“安全”命令。选择安全邮件的“高级”选项进行设置。

发送签名的电子邮件。启动 Outlook Express 6.0，单击“新邮件”，撰写新邮件。同时选中工具中的“数字签名” 选项。单击“发送”按钮，签名邮件发送成功。当收件人收到并打开有数字签名的邮件时，将看到数字签名邮件的提示信息，单击“继续”按钮后，才可阅读到

该邮件的内容。

接收签名电子邮件。若邮件在传输过程中被他人篡改或发信人的数字证书有问题，将出现“安全警告”提示。收到邮件后，可以看到邮件的右边中间有一个红色的小图标，单击它，可以看到相关的数字证书信息。

发送加密的电子邮件。发送加密邮件前必须先获得接收方的数字标识，可以首先让接收方发一份签名邮件来获取对方的数字标识、或者直接到认证中心的网站去查询下载来获取对方的数字标识、或者通过附件传送证书（扩展名.cer）。启动 Outlook Express 6.0，单击“新邮件”，撰写新邮件，同时选中工具的“加密”选项。

接收加密的电子邮件。接收到加密邮件后需要解密。接收加密的电子邮件，确定后即可。

4. 保护网站以防假冒

为了防范黑客假冒网站，可以申请一个服务器证书（例如到广东省电子商务认证中心 http://www.cnca.net/申请），然后双击“控制面板|管理工具|Internet 信息服务|默认 Web 站点|目录安全性|服务器证书”，按提示安装服务器证书。安装成功后，在网站醒目位置将显示“VeriSign 安全站点”签章、并提示用户点击验证此签章。点击此签章，就会连接 VeriSign 全球数据库验证网站信息，然后显示真实站点的域名信息及该站点服务器证书的状态。另外，如果网站有以下两样标志，则表明该网站激活了服务器证书，此时已建立了 SSL 连接。观察要提交个人信息页面的网址前，是否带有“https://”标志（这里的 S 代表安全网站）；观察网页的状态栏中是否有一把金色小锁，双击该锁会弹出该站点的服务器证书，里面包含了真实站点的域名及证书有效期。如果状态栏中没有金锁、或者它是打开的，那么表示提交的信息将不会被加密。

5. 使用代码签名防止软件被篡改

可以利用代码签名证书给软件签名，防止软件被篡改（例如软件中添加木马或病毒）。微软 Authenticode 工具包中有一个 signcode.exe，可以专门给代码签名，能对 32 位的、后缀为.exe（PE 文件）、.cab、.dll、.vbd 和.ocx 的文件进行数字签名。signcode 运行后会出现数字签名向导，首先选择要签名的文件，下一步后会出现签名选项，一种是典型，一种是自定义。选择“典型”，适用于代码签名证书存储在 IC 卡和浏览器等存储器中。选择“自定义”，适用于代码签名证书以 SPC 文件形式存储，私钥文件以 PVK 形式存储（Authenticode 工具包中 cert2spc.exe 将 cer 格式证书转换成 spc 格式证书，即 PKCS #7 证书）。完成签名后，可以右击经过数字签名的代码文件，然后在弹出的快捷菜单中选择“属性”命令，查看其签名和证书，即可得知该软件的发布者。

习 题 9

1. 设计一个安全的电子邮件系统应从哪几个方面进行考虑？
2. 安全电子邮件标准有哪些？

3. 一个完整的电子邮件系统包括哪几个组件？
4. 电子邮件主要存在哪些安全问题？
5. 简述 PGP 的功能。
6. PGP 如何实现加密和认证？
7. S/MIME 中增强了哪些安全服务？
8. 简单描述一下 S/MIME 的应用。

第 10 章　IP 安全

Internet 的网络安全体系结构目前仍处在发展阶段，人们正在从不同的角度来试图解决 Internet 的安全问题。网络层是 TCP/IP 协议网络中最关键的协议层，IP 作为网络层协议，其安全机制可对其上层的各种应用服务提供透明的覆盖式安全保护。因此，IP 安全是整个网络安全的基础，是网络安全的核心。

10.1　概　　述

IP 层是 TCP/IP 网络中最关键的一层，IP 作为网络层协议，其安全机制可对它上层的各种应用服务提供透明的覆盖式安全保护。为了实现安全 IP，Internet 工作任务组 IETF 于 1994 年开始了一项 IP 安全工程，专门成立了 IP 安全协议工作组 IESEC，来制定和推动一套称为 IPSec(Internet Protocol Secutity)的 IP 安全协议标准。其目标就是把安全特征集成到 IP 层，以便对 Internet 的安全业务提供低层的支持。

10.1.1　IPSec 的结构

IPv4 包本身没有任何安全特性，攻击者很容易便可伪造出 IP 包的地址、修改其内容、重播以前的包以及在传输途中拦截并查看包的内容。因此，有时收到的 IP 数据报源地址可能不是来自真实的发送方(IP 头内的源地址)；包含的原始数据可能遭到更改；原始数据在传输中可能被其他人浏览。

IPSec 的目标是为 IPv4 和 IPv6 提供透明的安全服务，保护网络通信免遭窃听和篡改，在 IP 层上提供数据起源的认证、无连接数据完整性认证、数据内容的机密性认证、抗重放保护和有限业务流机密性保证等安全服务。各种应用程序可以享用 IP 层提供的安全服务和密钥管理，而不必设计和实现自己的安全机制，因此减少了密钥协商的开销，也降低了产生安全漏洞的可能性。

IPSec 可以保障主机之间、网络安全网关(如路由器或防火墙)之间或主机与安全网关之间的数据包的安全。它主要提供认证(Authentication)和保密(Confidentiality)两种功能。

使用 IPSec 可以防范以下几种网络攻击。

(1) Sniffer：IPSec 对数据进行加密以对抗 Sniffer，保持数据的机密性。

(2) 数据篡改：IPSec 用密钥为每个 IP 包生成一个消息认证码(MAC)，该密钥仅由数据的发送方和接收方共享。对数据包的任何篡改，接收方都能够检测到。保证了数据的完整性。

(3) 身份欺骗：IPSec 的身份认证机制实现了数据起源认证。

(4) 重放攻击：IPSec 防止了数据包被捕获并重新投放到网上，接收方会检测并拒绝老

的或重复的数据包；IPSec 通过序列号实现了抗重放攻击。

(5) 拒绝服务攻击：IPSec 依据 IP 地址范围、协议，甚至特定的协议端口号来决定哪些数据流需要受到保护，哪些数据流可以被允许通过，哪些需要拦截。

IPSec 对于 IPv4 是可选使用的，对于 IPv6 是强制使用的。IPSec 规范中包含大量的 RFC 文档，主要包括：IP 安全体系结构 SA(Security Architecture for the Internet Protocol，RFC2401)、IP 认证头 AH(Authentication Header，RFC2402)、IP 封装安全载荷 ESP(IP Encapsulating Security Payload，RFC2406)、Internet 密钥交换(The Internet Key Exchange，RFC2409)、认证和加密算法。IPSec 安全体系结构如图 10-1 所示。

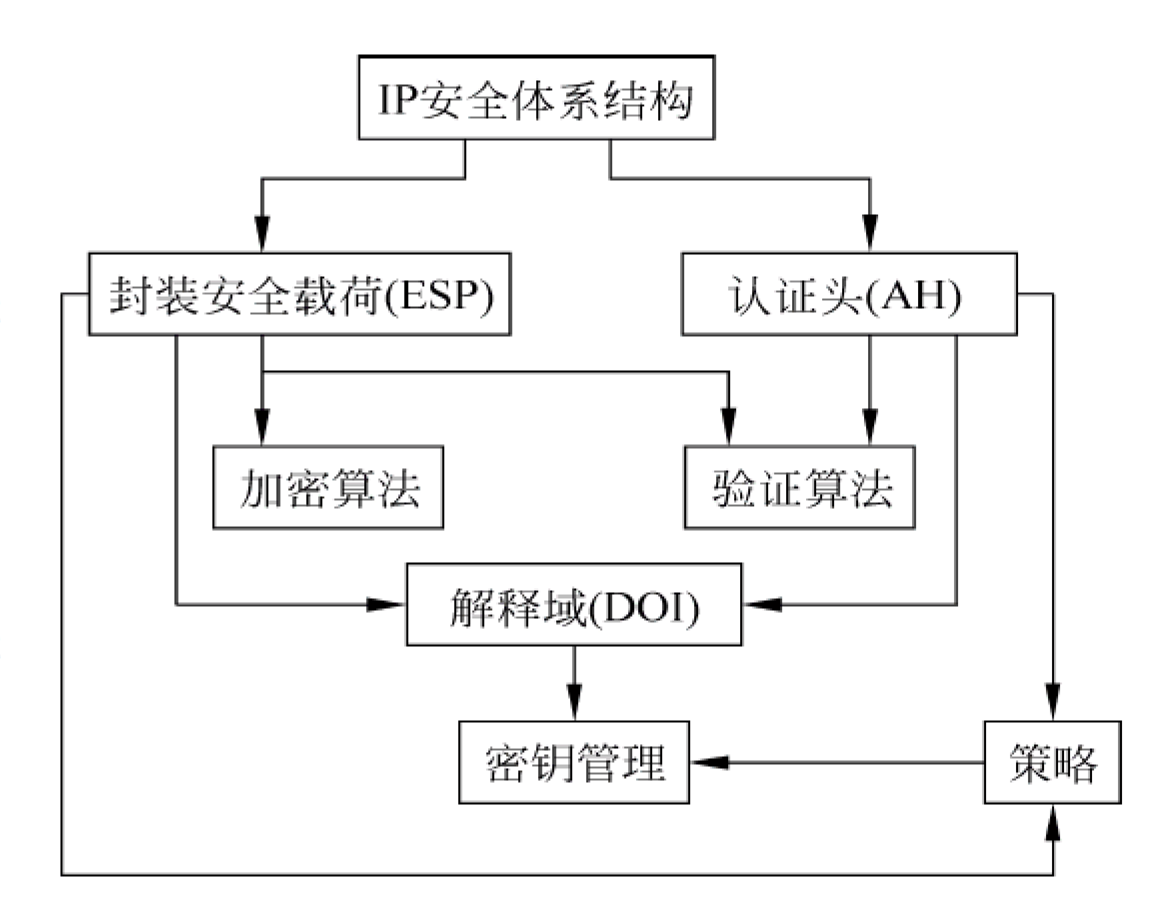


图 10-1 IPSec 安全体系结构

IP 安全体系结构：系统描述了 IPSec 的工作原理、系统组成以及各个组件是如何协同工作提供上述安全服务的，是关于 TCP/IP 协议簇总的概述。它包含了 IP 安全技术的一般概念、安全目标、安全服务、定义和定义 IPSec 的技术机制。

- 封装安全载荷(ESP)：覆盖了为包加密(可选身份验证)与 ESP 的使用相关的包格式和常规问题。
- 验证头(AH)：包含使用 AH 进行包身份验证相关的包格式和一般问题。
- 加密算法：描述各种加密算法如何用于 ESP。
- 认证算法：描述各种身份认证算法如何用于 AH 和 ESP。
- 密钥管理：密钥管理的一组方案，其中 IKE 是默认的密钥自动交换协议，IKE 适合为任何一种协议协商密钥，并不仅限于 IPSec 的密钥协商，协商的结果通过解释域转化为 IPSec 所需的参数。
- 解释域(Domain Of Interpretation，DOI)：为了 IPSec 通信两端能相互交互，通信双方应该理解 AH 协议和 ESP 协议载荷中各字段的取值，因此通信双方必须保持对通信消息相同的解释规则，即应持有相同的解释域。IPSec 至少已经给出了两个解释域，IPSec DOI 和 ISAKMP DOI，它们各有不同的使用范围。解释域定义了协议用来确定安全服务的信息通信双方必须支持的安全策略，规定所采用的句法，命名相关安全服务信息时的方案，包括机密算法、密钥交换算法、安全策略特性和认证中心等。
- 策略：决定两个实体之间能否通信，以及如何进行通信。策略部分是唯一尚未成为标准的组件。

每一种安全保护方式都有其特殊的安全特性，IPSec 也不例外，下面介绍的是 IPSec 安全协议的五个特性。

(1) 不可否认性：该特性可以证实数据发送方是唯一可能的发送者，发送者不能否认发送过数据。在数据过程中，使用 IPSec 技术时，在验证时使用的是公钥技术，发送方用私钥产生一个数字签名随数据一起发送，接收者用发送者的公钥来验证数字签名是否被修改。

(2) 反重播性：为确保每个IP包的唯一性，保证数据万一被截取复制后，不能再被重新利用、重新传输回目的地址所使用的技术。

(3) 数据完整性：防止传输过程中数据被篡改，确保发出数据和接收数据的一致性。IPSec利用Hash函数为每个数据包产生一个加密校验和，接收方在打开包前先计算校验和，若包被篡改导致检查的两个校验和不相符，数据包即被丢弃。

(4) 数据可靠性：在数据传输前，要对传输的数据进行加密处理，可以保证在传输过程中，即使数据包被截取，数据也无法被读出。

(5) 认证：数据源发送身份数据信息给接收方，由接收方检查数据是否被修改并验证发送者身份的合法性，只有通过认证的系统才可以建立通信连接。

10.1.2 传输模式与隧道模式

IPSec协议(包括AH和ESP)既可用来保护一个完整的IP载荷，亦可用来保护某个IP载荷的上层协议。这两方面的保护分别是由IPSec两种不同的模式来提供的。其中，传输模式(传送模式)用来保护上层协议；而隧道模式(通道模式)用来保护整个IP数据报。两种IPSec协议(AH和ESP)均能同时以传送模式或隧道模式工作。

传输模式(如图10-2所示)主要用于对IP包中的上层协议的保护。在IPv4中，传输模式的IPSec头插入到IP报头之后、高层传输协议(如TCP、UDP)之前。在IPv6中，该模式的IPSec头出现在IP头及IP扩展头之后、高层传输协议之前。传输模式典型地用于两个主机的端到端通信。以传输模式运行的ESP协议对负载进行加密和认证(认证性可供选择)，但不对IP报头进行加密和认证。以传输模式运行的AH协议对负载及报头中选择的一部分进行认证。

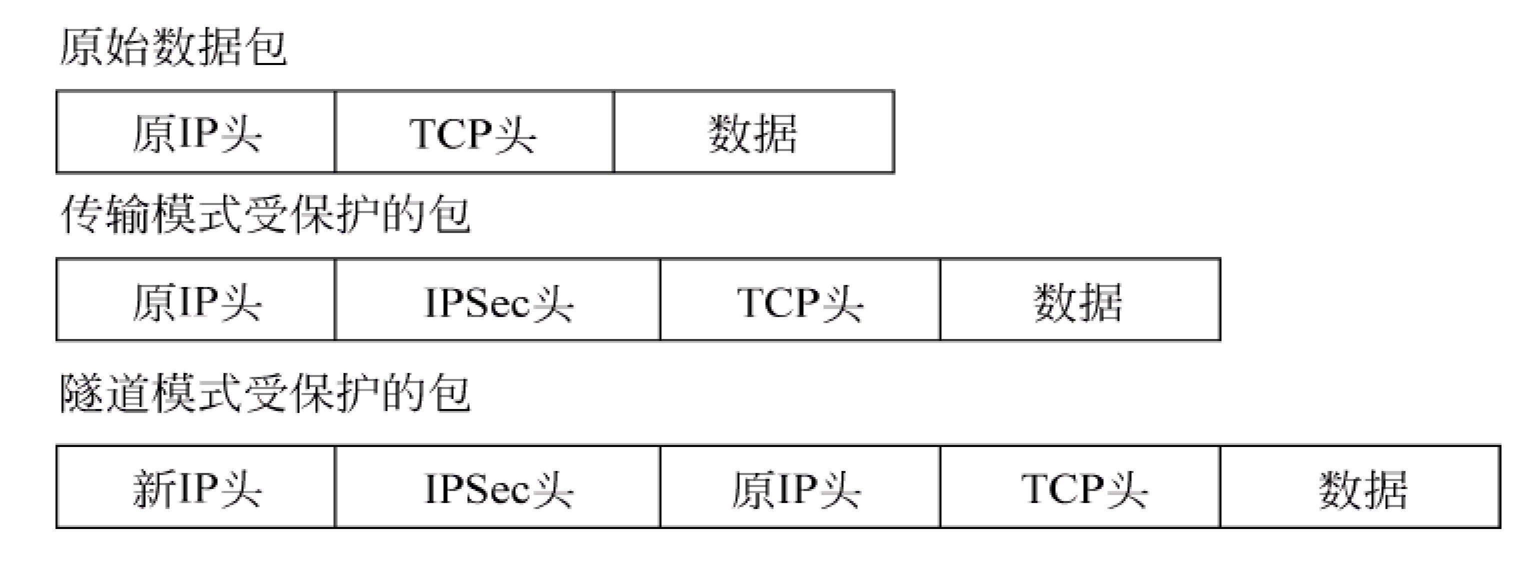


图10-2 传送模式与隧道模式保护的数据包

隧道模式(如图10-2所示)用于对整个IP数据报的保护，它是将一个数据报用一个新的数据报包装，即给原数据报加一个新的报头，称为外部报头，这样原数据报就成为新数据报的负载。因此原数据报在整个传送过程中就像在隧道中一样，传送路径上的路由器都无法看到原数据报的报头。由于封装了原数据报，新数据报的源地址和目标地址都与原数据报不同，从而增加了安全性。隧道模式下，要保护的整个IP包都封装到另一个IP数据报里，同时在外部与内部IP头之间插入一个IPSec头。外部IP头指明进行IPSec处理的目的地址，内部IP头指明最终的目的地址。若构成一个安全关联的两个终端中至少有一个是安全网关(而不再是主机)，则这个安全关联就必须采用隧道模式。在隧道模式下，IPSec报文要进行分段和重组操作，并且可能要再经过多个安全网关才能到达安全网关后面的目的主机。

例如，主机 A 产生一个数据报，其目标地址是另一网络中主机 B 的地址，A 将自己的数据报发送给 A 所在网络的防火墙 C。防火墙 C 过滤自己将发送的所有数据报以决定哪些需要进行 IPSec 处理，如果 A 发往 B 的数据报需要 IPSec 处理，防火墙 C 则对其进行处理并用一个新报头对其封装，新报头中源地址为防火墙 B 的 IP 地址，目标地址是 B 所在局域网中防火墙 D 的 IP 地址。数据报在到达防火墙 D 之前，所有中间路由器仅检查外部报头，到达防火墙 D 后，则剥去外部报头并将原数据报发给 B。

以隧道模式运行的 ESP 协议对整个内部数据报（包括报头）加密和认证（认证性可供选择），而 AH 协议则对整个内部数据报和外部报头中的所选部分进行认证。

10.1.3　安全关联

1. 安全关联的定义

安全关联（Security Association，SA）又称为“安全联盟”，是两个应用 IPSec 实体（主机、路由器）间的一个单向逻辑连接，决定保护什么、如何保护以及谁来保护通信数据。

一个 SA 可由三个参数唯一地表示为：

＜安全参数索引，目标 IP 地址，安全协议标识符＞

三个参数的含义分别为：

（1）安全参数索引（Security Parameters Index，SPI）：赋值给该 SA 的 32 位比特串，其位置在 AH 和 ESP 报头中，用于标识有相同的目标地址和相同安全协议的不同 SA。

（2）目标 IP 地址：即 SA 中接收方的 IP 地址，该地址可以是终端用户系统或防火墙、路由器等网际设备的地址。目前的 SA 管理机制只支持单目地址。

（3）安全协议标识符：表示 SA 使用的协议是 AH 协议还是 ESP 协议。

一个 SA 可以处于两种模式之一：传输模式或隧道模式，这取决于 SA 中使用的协议的模式。

SA 是单向的，所以两个系统之间的双向通信需要两个 SA，每个方向一个。

一个 SA 对 IP 数据报不能同时提供 AH 和 ESP 保护。有时，特定的安全策略要求对通信提供多种安全保护，这就需要使用多个 SA。当把一系列 SA 应用于业务流时，称为 SA 束。SA 束的顺序由安全策略决定，SA 束中各个 SA 的终点可能不同。例如，一个 SA 可能用于移动主机与安全网关之间，而另一个可能用于移动主机与安全网关内的主机。

2. 安全关联的创建

SA 既可人工创建，亦可采用动态创建方式。

若用人工方式创建，安全参数由管理员按安全策略人工指定、人工维护。但是，人工维护容易出错，而且人工建立的 SA 没有存活时间的说法，除非再用人工方式将其删除，否则便会一直存在下去。

若用动态方式创建，则 SA 有一个存活时间与其关联在一起。这个存活时间通常是由密钥管理协议在 IPSec 通信双方之间加以协商而确立下来的，存活时间（TTL）非常重要。若超时使用一个密钥，会为攻击者侵入系统提供更多的机会。SA 的自动建立和动态维护是通过 IKE 进行的。如果安全策略要求建立安全、保密的连接，但却不存在相应的 SA，IPSec 的内核则启动或触发 IKE 协商。

3. 与安全关联有关的数据库

IPSec 的实现需要维护两个与 SA 有关的数据库：安全关联数据库(Security Association Database,SAD)和安全策略数据库(Security Policy Database,SPD)。

SAD 包含每一个 SA 的参数信息,例如 AH 或 ESP 算法和密钥、序列号、协议模式以及 SA 生命期。对于外界(Outbound)数据的处理,会有一个 SPD 数据项中包含指向某个 SAD 数据项的指针。SPD 决定了一个给定的数据包究竟使用哪个 SA。对于界内(Inbound,带有 IPSec 头)数据的处理,由 SAD 来决定如果处理。

SAD 中包含以下字段。

序号计数器：32 比特,用于产生 AH 或 ESP 头的序号,仅用于流出数据包。

序号计数器溢出标志：标识序号计数器是否溢出。如溢出,则产生一个审计事件,并禁止用 SA 继续发送数据包。

抗重播窗口：32 比特计数器,用于决定流入的 AH 或 ESP 数据包是否为重发。仅用于流入数据包,如接收方不选择抗重播服务(如手工设置 SA 时),则抗重播窗口未被使用。

AH 信息：表示 AH 所使用的认证算法、密钥、密钥生命期等。

ESP 信息：表示 ESP 所使用的加密认证算法、密钥、初始值、密钥生命期等。

SA 的生命期：一个时间间隔。超过这一间隔后,应建立一个新的 SA(以及新的 SPI)或终止通信。生存期以时间或字节数为标准,或将两者结合使用,并优先采用先到者。

IPSec 协议模式：隧道、传输或混合方式(通配符),说明应用 AH 或 ESP 的模式。

路径最大传输单元 MTU：指不被分段而能传输的最大数据报长度。

其中 AH 信息和 ESP 信息分别仅为 AH 协议和 ESP 协议所要求,其他参数在两种协议中都被要求。

10.1.4 IPSec 安全策略

IPSec 的基本架构定义了用户能以多大的精度来设定自己的安全策略。某些通信可以为其设置某一级的基本安全措施；而对其他通信则可为其应用完全不同的安全级别。如可在一个网络安全网关上制定 IPSec 策略,对在其本地保护的子网与远程网关的子网间通信的所有数据,全部采用 DES 加密,并用 HMAC-MD5 进行验证；另外,从远程子网发给一个邮件服务器的所有 Telnet 数据均用 3DES 进行加密,同时用 HMAC-SHA 进行验证；最后对于需要加密的、发给另一个服务器的所有 Web 通信数据,则用 IDEA 满足其加密要求,同时用 HMAC-RIPEMD 进行验证。

IPSec 本身没有为策略定义标准,目前只规定了两个策略组件：SAD(安全关联数据库)和 SPD(安全策略数据库)。在 IPSec 系统中,IPSec 策略由 SPD 加以维护。在 SPD 这个数据库中,每个条目都定义了要保护的是什么通信、怎样保护它以及和谁共享这种保护。策略描述主要包括两方面的内容：一是对保护方法的描述,二是对通信特性的描述。

对保护方法的描述：对于进入或离开 IP 堆栈的每个包,都必须检索 SPD 数据库,调查可能的安全应用。对一个 SPD 条目来说,它可能定义了下述几种行为：丢弃、绕过以及应用。其中,“丢弃”表示不让这个包进入或外出；“绕过”表示不对一个流出的包应用安全服务,也不指望一个流入的包进行了安全处理；而“应用”是指对流出的包应用安全服务,同时要求流入的包已应用了安全服务。对那些定义了“应用”行为的 SPD 条目,它们均会指向一个或一套 SA,表示要将其应用于数据包。

使用选择符描述通信特性。IPSec 通信到 IPSec 策略的映射关系是由“选择符(Selector)”来建立的。选择符标识通信的一部分组件,它既可以是一个粗略的定义,也可以是一个非常细致的定义。IPSec 选择符包括 6 方面内容:目标 IP 地址、源 IP 地址、名字、上层协议、源和目标端口以及一个数据敏感级(假如也为数据流的安全提供了一个 IPSec 系统)。

- 目的 IP 地址:可为单个 IP 地址、地址列表、地址范围或通配(掩码)地址。后两种用于支持共享一个 SA 的多个目的系统。
- 源 IP 地址:可为单个 IP 地址、地址列表、地址范围或通配(掩码)地址。后两种用于支持共享一个 SA 的多个源发系统。
- 名字:名字字段用于标识与一名有效用户或者系统名称关联在一起的策略。其中包括一个 DNS 名、X.500 区分名或者在 IPSec DOI 中定义的其他名字类型。只有在 IKE 协商期间(而非包处理期间),名字字段才能作为一个选择符使用。在包处理期间,这一字段不能作为选择符使用。因为目前无法把一个 IP 地址和一个名字结合在一起。
- 传输层协议:协议字段指定了传输协议(意思是只有传输协议可以访问)。许多情况下,只要使用了 ESP,传送协议无法访问,这时需使用通配符。
- 源和目标端口:TCP 或 UDP 端口号,可为单个端口、端口列表或通配端口。如果端口不能访问,则要使用通配符。
- 数据敏感等级:通信数据的保密等级,可分为普通、秘密、机密、绝密。

IPSec 结构定义了 SPD 和 SAD 这两种数据库之间如何沟通。这是通过 IPSec 处理功能——封装与拆穿来实现的。

对于流出数据包,IPSec 协议要先查询 SPD,确定数据包应使用的安全策略。如果检索到的数据策略是应用 IPSec,再查询 SAD(每个 SPD 的元组都有指针指向相关的 SAD 的元组),确定是否存在有效的 SA。

(1) 若存在有效的 SA,则取出相应的参数,将数据包封装(包括加密、验证,添加 IPSec 头和 IP 头等),然后发送。

(2) 若尚未建立 SA,则启动或触发 IKE 协商,动态地创建 SA,协商成功后按(1)中步骤处理,不成功则应将数据包丢弃,并记录出错信息。

(3) 存在 SA 但无效,将此信息向 IKE 通告,请求协商新的 SA,协商成功后按(1)中的步骤处理,不成功则应将数据包丢弃,并记录出错信息。

对于流入数据包,IPSec 通过包头信息包含的 IP 目的地址、IP 安全协议类型(AH 或 ESP)和 SPI 在 SAD 中查找对应的 SA。如得到有效的 SA,则对数据包进行解封(还原),再查询 SPD,验证为该数据包提供的安全保护是否与策略配置的相符。如相符,则将还原后的数据包交给 TCP 层或转发。如不相符、或要求应用 IPSec 但未建立 SA、或 SA 无效,则将数据包丢弃,并记录出错信息。

10.2　封装安全载荷

封装安全载荷(ESP)属于 IPSec 的一种协议,ESP 提供机密性、数据起源认证、无连接的完整性、抗重播服务和有限业务流机密性。ESP 本身是一个 IP 协议,协议号是 50。

10.2.1 封装安全载荷包格式

不管 ESP 处于什么模式，ESP 头都会紧紧地跟在一个 IP 头之后，在 IPv4 中，ESP 头紧跟在 IP 头后面(其中包括任意选项)。在 IPv6 中，ESP 头的放置与扩展头是否存在有关。ESP 头肯定插在扩展头(它可改变源到目的主机之间的路由)之后。其中包括逐跳跃、路由选择和分段头。

ESP 头包含下面一些字段(如图 10-3 所示)。

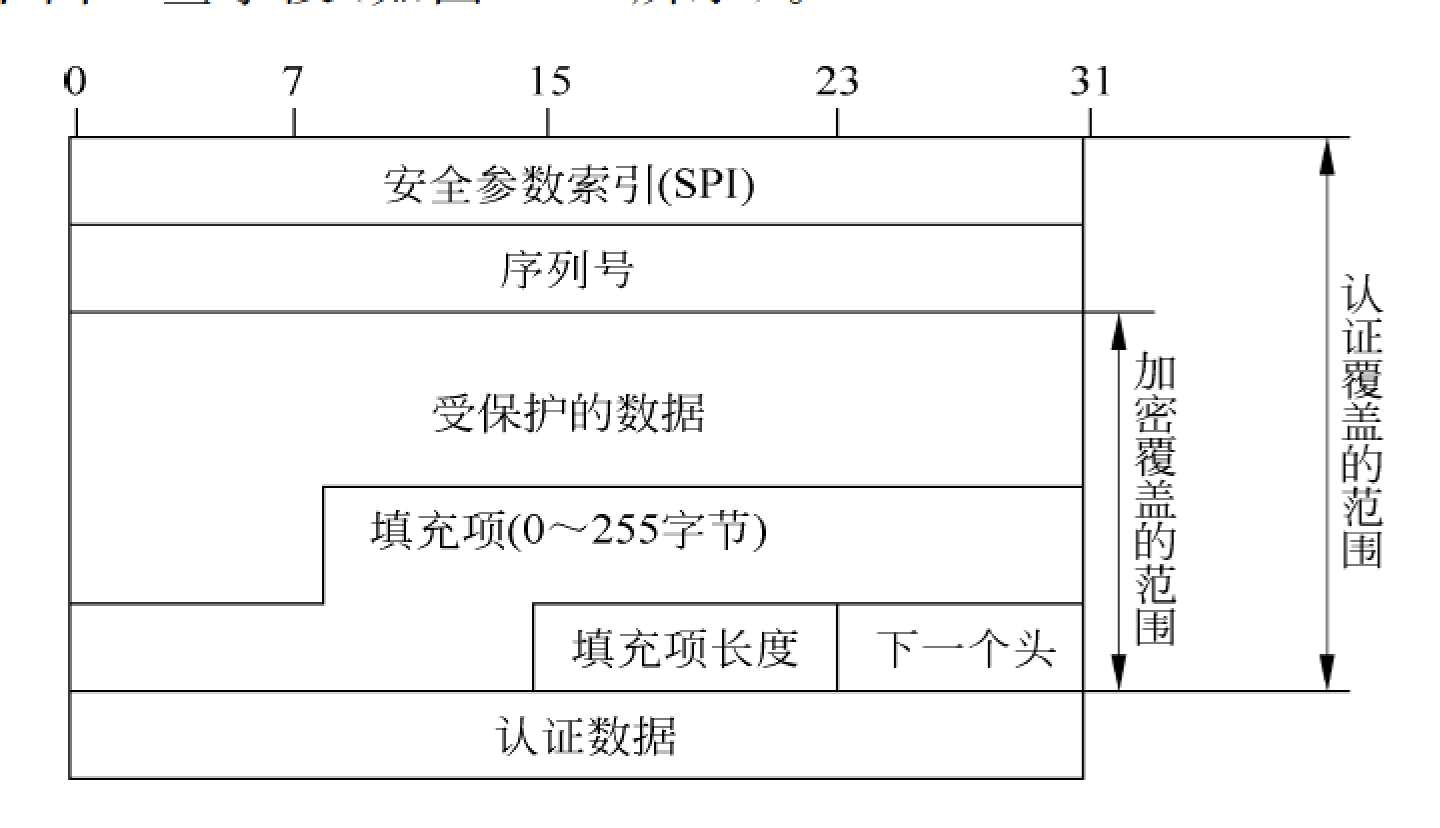


图 10-3 ESP 头

- 安全参数索引 SPI(32 位)：SPI 是个伪随机数，和外部 IP 头的目标地址以及安全协议结合在一起，用来标识用于处理数据包的特定的那个安全关联。
- 序列号(32 位)：序列号是一个独一无二的、单向递增的、并由发送端插在 ESP 头的一个号码。发送方的计数器和接收方的计数器在一个 SA 建立时被初始化为 0，使用给定 SA 发送的第一个分组的序列号为 1，如果激活抗重播服务(默认地)，传送的序列号不允许循环。因此，在 SA 上传送第 2^{32} 个分组之前，发送方计数器和接收方计数器必须重新置位(通过建立新 SA 和获取新密钥)，序列号使 ESP 具有了抵抗重播攻击的能力。
- 受保护数据(可变)：通过加密保护的传输层协议内容(传输方式)或 IP 包(通道模式)。因为 IP 包可能会失序到达，因此每个 IP 包必须携带能使接收者进行解密所要求的密码同步数据(如初始化向量 IV)。初始化向量 IV 可以在受保护数据字段的开头携带，并且 IV 通常不加密，但经常被看做是密文的一部分。
- 填充(0～255 字节)：主要用于加密算法要求明文使某个数目字节的倍数、保证填充长度字段和下一个头字段排列在 32 位字的右边、提供部分业务流机密性。用于在 ESP 中保证边界的正确。
- 填充长度(8 位)：指出填充字节的数目。填充长度字段是硬性规定的，因此，即使没有填充，填充长度字段仍会将它表示出来。
- 下一个头(8 位)：标识受保护数据的第一个头。例如，IPv6 中的扩展头或者上层协议标识符。此字段表明数据类型。
- 认证数据(可变)：完整性检查值。认证数据是可变长字段，它包含一个完整性校验值(ICV)，ESP 分组中该值的计算不包含认证数据本身。字段长度由选择的认证算

法指定。认证数据字段是可选的，只有 SA 选择认证服务，才包含认证数据字段。认证算法规范必须指定 ICV 长度、验证规则和处理步骤。

ESP 所采用的加密算法由 SA 指定。为提高加密效率，ESP 使用对称密码算法（如 DES 算法、3DES 算法）。由于机密性是可选择的，因此加密算法可以是“空”。RFC1829 中指定的 ESP 加密算法是 DES-CBC。

用于计算完整性校验值（ICV）的认证算法由 SA 指定。对于点到点的通信，合适的鉴别算法包括基于对称密码算法（如 DES）或基于单向 Hash 函数（如 MD5 或 SHA-1）的带密钥的消息认证码（MAC）。RFC1828 建议的认证算法是带密钥的 MD5。由于认证算法是可选的，因此此算法可以是“空”。加密算法和认证算法虽然都可以为空，但两者不能同时为空。

10.2.2　封装安全载荷协议处理

在 ESP 的传输模式中，ESP 协议将上层协议数据作为 ESP 封装的载荷数据，而原 IP 包头仍作为封装后的 IP 包的包头。这种封装模式保留了原 IP 包头信息，只是 IP 包头的协议字段值变为 50，表示 IP 包头后紧接的载荷为 ESP 载荷。而 IP 包头的协议字段的原有值被记录于 ESP 载荷的下一个字段。ESP 头插在 IP 包头之后和一个上层协议（如 TCP、UDP、ICMP 等）之前。其封装格式如图 10-4 所示。

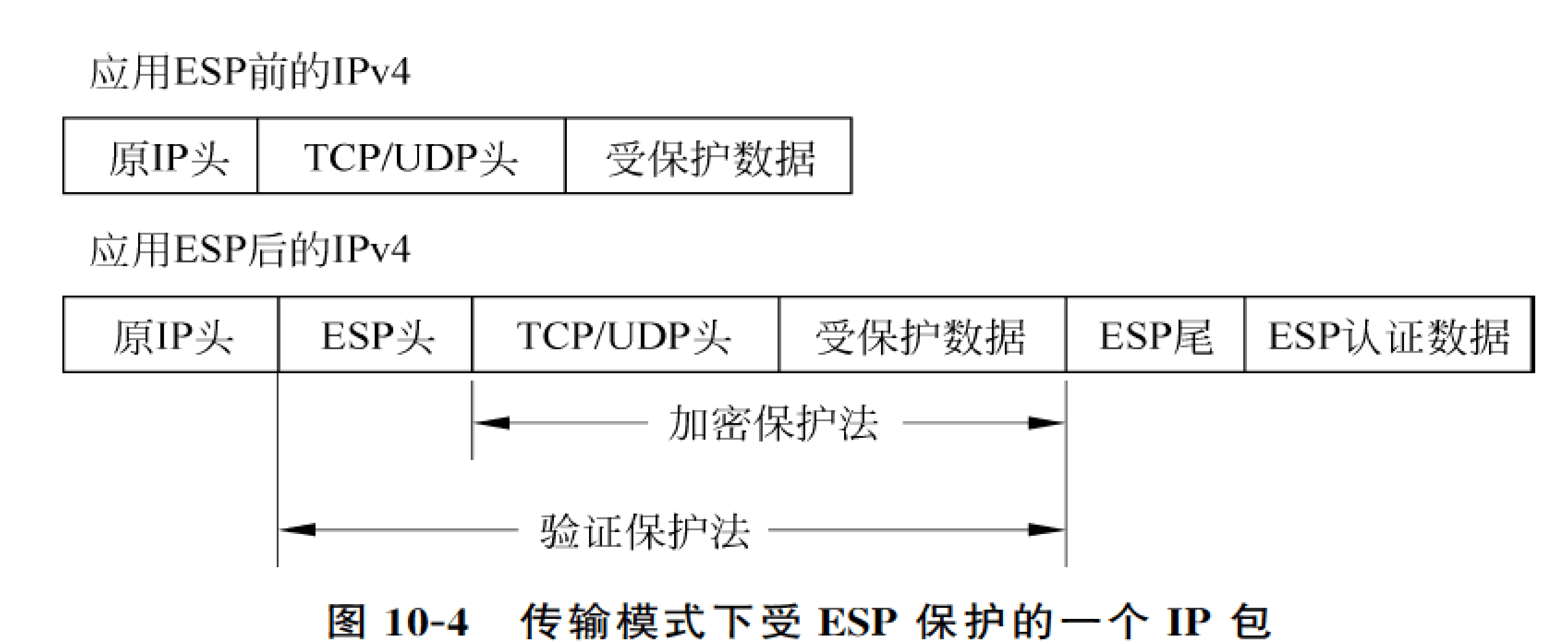


图 10-4　传输模式下受 ESP 保护的一个 IP 包

在隧道模式，整个 IP 包（包括原 IP 包头）都被封装在 ESP 有效载荷中，并产生一个新 IP 包头附着在 ESP 头之前。这个新的 IP 包头的协议字段将是 50，以表明 IP 包头之后是一个 ESP 头。ESP 的隧道模式对整个 IP 包提供安全保护（机密性和完整性），既可用于主机也可用于安全网关。当 ESP 在安全网关上实现时，必须采用隧道模式。其封装格式如图 10-5 所示。

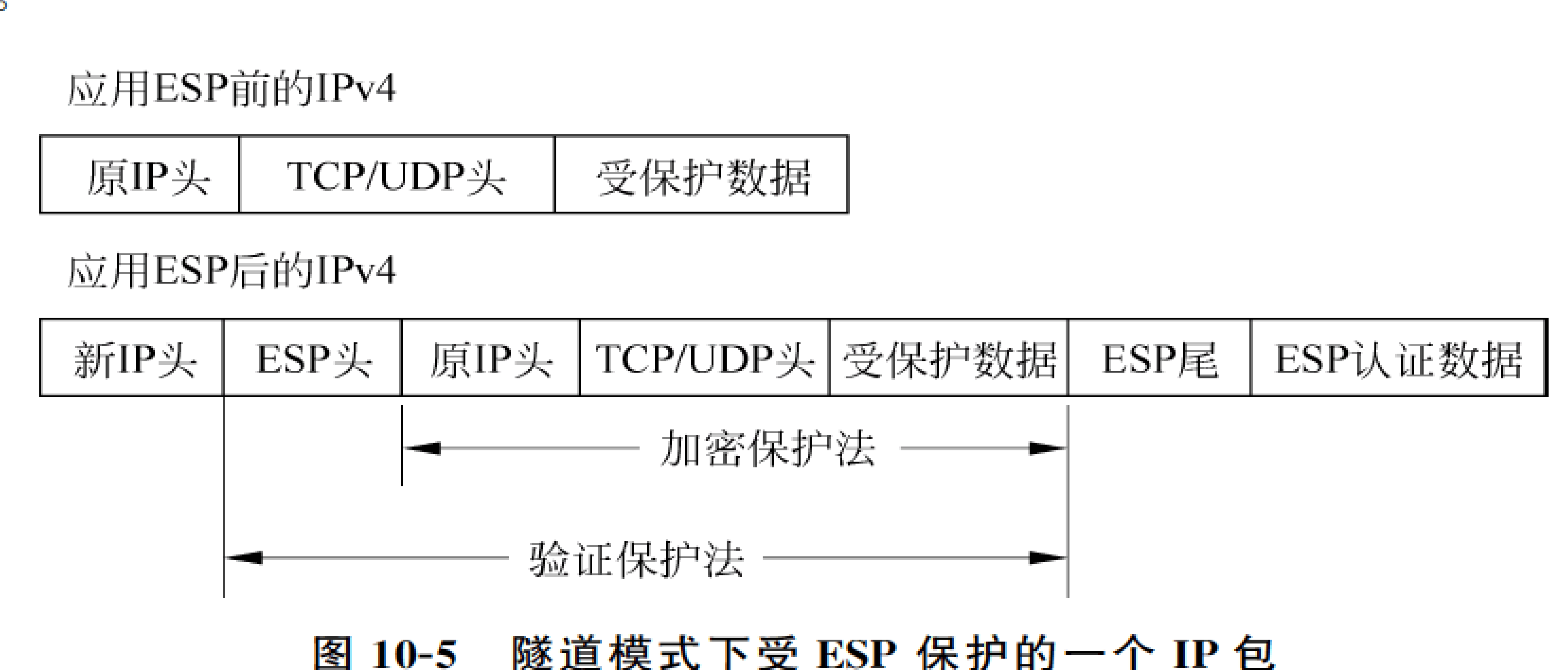


图 10-5　隧道模式下受 ESP 保护的一个 IP 包

由于ESP同时提供了机密性以及认证机制，所以在其SA中必须同时定义两套算法。用来确保机密性的算法叫做cipher(加密器)，负责认证的叫做authenticator(认证器)。

1. 流出分组的处理

对在IPv4上运行的传输模式应用来说，ESP头紧跟在IP头(包括IP头可能有的任何选项)之后，IP头的协议字段被复制到ESP头的“下一个头”字段中，ESP头的其余字段则被填满。SPI字段分配到的是来自SAD的、用来对这个包进行处理的特定SA的SPI；填充序列号字段的是序列中的下一个值；填充数据会被插入，其值被分配；同时分配的还有填充长度值。随后，IP头的协议字段得到的是ESP的值50。除了头插入位置不同之外，IPv6处理规则基本上类似于IPv4。ESP头可插在任意一个扩展头之后。

对隧道模式应用来说，ESP头是加在IP包前面的。如果封装的是一个IPv4包，那么ESP头的下一个头字段分配到值4；如果封装的是一个IPv6包，则分配到值41。其他字段的填充方式和在传输模式中一样。随后，在ESP头的前面新增了一个IP头，并对相应的字段进行填充(赋值)——源地址对应于应用ESP的那个设备本身；目标地址取自于用来应用ESP的SA；协议号赋值为50；其他字段的值则参照本地的IP处理加以填充。

不管哪种模式下，接下去的步骤都是相同的。从恰当的SA中选择加密器(加密算法)，对包进行加密(从载荷数据的开头，一直到“下一个头”字段)。随后，使用恰当的SA中的认证器，对包进行认证(自ESP头开始，中间经过加密的密文，一直到ESP尾)。随后，将认证器的结果插入ESP尾的“认证数据”字段中。

对外出数据包进行处理的最后一步是：重新计算位于ESP前面的IP头的校验和。

注意在添加ESP头时，不必进行分段检查。如果包(在已采用ESP之后)大于它流经的那个接口的MTU，只好对它进行分段。这和一个完整的ESP包离开该设备，并在网络中的某个地方被分成段没有什么区别。

2. 进入分组的处理

接收端在收到一个ESP包之后，若不对这个包进行处理，就无法得知它究竟处于隧道模式，还是传输模式。根据对这个包进行处理的SA，便可知道它到底处在什么模式下。但除非完成了对它的解密，实际上不可能知道ESP保护的是什么。

如果收到的IPSec包是一个分段，必须把它保留下来，直到这个包的其他部分收完为止。即在ESP处理之前进行重组。

收到一个(已重组的)包含ESP头的包时，根据目的IP地址、安全协议(ESP)和SPI，接收方确定适当的SA。SA指出序列号字段是否被校验(即是否选择抗重播)，认证数据字段是否存在，它将指定解密和ICV计算(如果适用)使用的算法和密钥。如果本次会话没有有效的SA存在(例如接收方没有密钥)，接收方必须丢弃分组，这是可审核事件。该事件的核查日志表项应该包含SPI的值、接收的日期/时间、源地址、目的地址、序列号和(IPv6)明文信息流ID。

一旦验证通过了一个有效的SA，就可用它开始包的处理。

检查序列号：序号字段是设计用来阻挡重放攻击的。当新的SA被建立的时候，发送者将序号计数器初始化为0。每当一个分组被发送给这个SA，发送者对这个计数器增1并将序号字段设置成计数器的值。这样，被使用的第一个值就是1。如果使用了反重放(默

认)，发送者不应该允许序号从 $2^{32}-1$ 循环到 0。否则，就有可能出现多个合法分组具有同样序号的情况。如果达到了 $2^{23}-1$ 的极限，发送者应该终止这个 SA 并且协商具有新密钥的新的 SA。因为 IP 是无连接、不可靠的服务，协议不能保证分组的按序交付，也不能保证所有的分组都会被交付。因此，IPSec 规定接收者应该实现大小为 W 的窗口，默认的情况下 W=64。窗口的右边界代表目前已经收到的合法分组的最高序号 N。对于任何已经正确接收的(即经过了正确鉴别的)其序号处于 N−W+1 到 N 范围之间的分组，窗口的相应插槽被标记。当收到一个分组时，按照如下步骤进行进入处理：

(1) 如果收到的分组落在窗口之内并且是新的，就进行 MAC 检查。如果分组被认证，就对窗口的相应插槽做标记。

(2) 如果收到的分组落在窗口的右边并且是新的，就进行 MAC 检查。如果分组被认证，那么窗口就向前走，使得该序号成为窗口的右边界，并对窗口的相应插槽做标记。

(3) 如果收到的分组落在窗口的左边，或者鉴别失败，就丢弃该分组，这是一个可核查的事件。

完整性校验值确认：如果选择验证，接收方采用指定的认证算法对 ESP 包计算 ICV(不包含认证数据字段)，确认它与认证数据字段中包含的 ICV 相同。如果计算得来的与接收的 ICV 匹配，那么数据包有效，可以被接收。如果验证失败，接收方将接收的 IP 数据包丢弃，这是可审核事件。

分组解密：通过取自 SA 的密钥和密码算法，对 ESP 包进行解密，从这个 ESP 包载荷数据开始之处到下一个头字段之间。

10.3　认　证　头

认证头(Authentication Header，AH)协议是 IPSec 体系结构中的一种主要协议，用于为 IP 数据包提供数据完整性、数据包源地址认证和一些有限的抗重播服务，它定义在 RFC 2402 中。AH 不提供对通信数据的加密服务，与 ESP 协议相比，AH 不提供对通信数据的加密服务，但能比 ESP 提供更加广的数据认证服务。

10.3.1　认证头的包格式

AH 是另一个 IP 协议，它的协议号是 51。紧跟在 AH 头部前面的协议头部(IPv4、IPv6 或扩展)的协议字段(IPv4)或下一头部字段(IPv6 或扩展)中包含 AH 的协议号是 51。AH 头格式如图 10-6 所示。

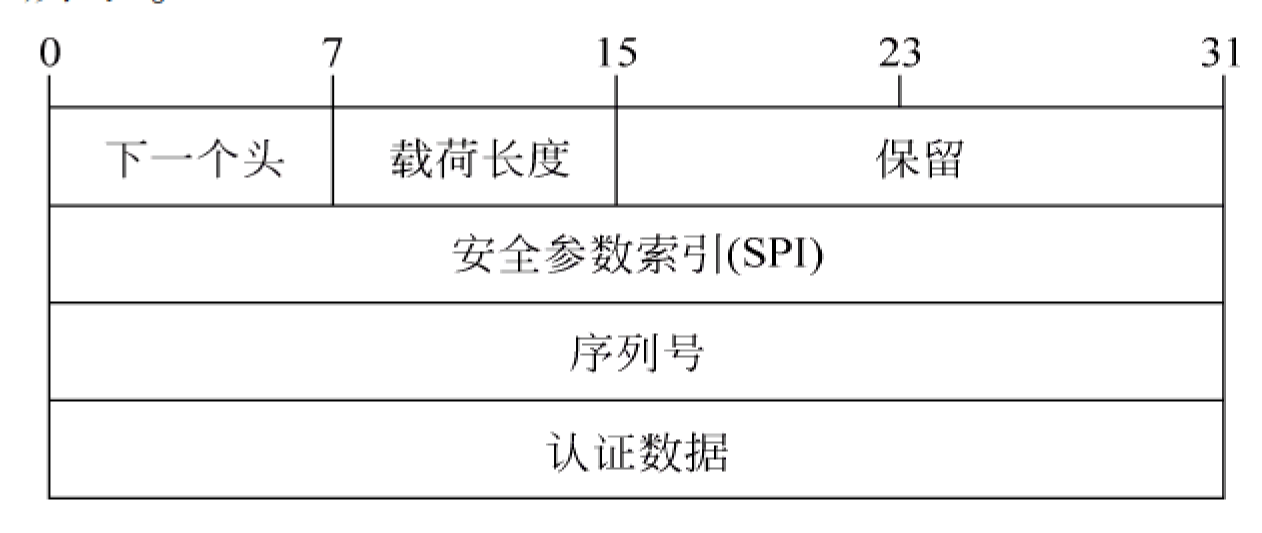


图 10-6　AH 头

- 下一个头(8 位):标识跟在认证头后的下一个头类型。在传输模式下,将是处于保护中的上层协议的协议号,比如 UDP 协议的值 17 或 TCP 协议的值 6。在隧道模式下,将是值 4(表示 IPv4)或 41(IPv6)。
- 载荷长度(8 位):在 32 位字中,这个字段包含 IPSec 协议头长度减 2 的值。协议头的固定部分是 96 位,即 3 个 32 位字。认证数据部分是可变长的,但有一个标准长度为 96 位,同样也是 3 个 32 位字。这样总共是 6 个 32 位字。最后减去 2 并进入载荷长度字段的值将是 4。
- 保留(16 位):为了将来使用。
- SPI(32 位):SPI 值 0 被保留来表明“没有安全关联存在”。
- 序列号(32 位):一个单向递增的计算器,等同于 ESP 中使用的序列号。序列号提供抗重播功能。当建立一个 SA 时,序列号被初始化为 0,使用该 SA 传递的第一个数据包的序列号就是 1。序列号不允许重复,这样每个 SA 上所能传递的 IP 报文的最大数目就是 $2^{32}-1$,当序列号达到最大值时,就需要建立一个新的 SA。
- 认证数据(可变):一个不固定的长度字段,也叫完整性校验值(ICV)或者 MAC,该字段长度必须是 32 位的整数倍。IPSec 规定必须支持 HMAC-SHA-1-96 和 HMAC-MD5-96。

用于计算完整性校验值(ICV)的认证算法由 SA 指定。AH 中的认证算法同 ESP 一样,只是验证所覆盖的范围不同。

10.3.2 认证头协议处理

和 ESP 一样,AH 可用于传送模式和隧道模式。不同之处在于它保护的数据一个是上层协议,一个是完整的 IP 数据报。任何一种情况下,AH 都要对外部 IP 头的固有部分进行身份认证。此外,AH 可被独立使用,或与 IP 封装安全负载(ESP)相结合使用,或通过使用隧道模式的嵌套方式。

AH 用于传输模式(如图 10-7 所示)时,保护的是端到端的通信。通信的终点必须是 IPSec 终点。这时 AH 头紧跟在原 IP 包的包头之后(和任意选项)和需要保护的上层协议数据(如 TCP/UDP 数据等)之前。这种模式的特点是保留了原 IP 包头信息。即信源/信宿地址不变,所有安全相关信息则都包含在 AH 头中。只是 IP 包头的协议字段由原来的值变为 51,表示 IP 包头后的紧接载荷为 AH 载荷;而 IP 包头的协议字段的原有值被记录于 AH 头的下一个头字段中。

计算认证数据时,输入如下:IP 数据包头(只包括在传输期间不变的字段或接受方可预测的字段,其余不定的字段全置为 0)、AH(除“认证数据”字段外其他的所有字段,“认证数据”字段被置为 0)、IP 数据包中的所有的上层协议数据。

AH 的隧道模式(如图 10-8 所示)将 AH 头插入原 IP 分组的 IP 包头之前,并在 AH 头之前添加新的 IP 包头。这个新的 IP 包头的协议字段将是 51,表明 IP 包头之后是一个 AH 头。

AH 的隧道模式对整个 IP 包提供鉴别保护。既可用于主机也可用于安全网关(安全路由器)。当 AH 在安全网关上实现时,必须采用隧道模式。这时,它将自己保护的 IP 包封装起来,并在 AH 头之前另加了一个新 IP 包头。内层 IP 包中包含了通信的原始目的地址,外层 IP 包则包含了 IPSec 端点(安全网关)的信宿地址。

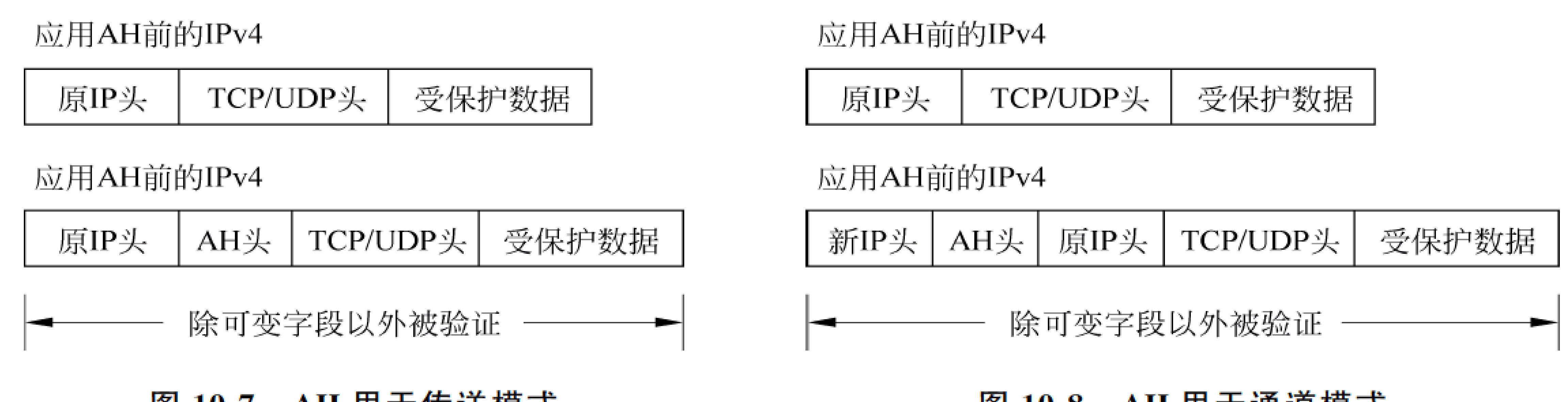


图 10-7　AH 用于传送模式　　**图 10-8　AH 用于通道模式**

1. 流出分组的处理

流出数据包与一个 SPD 条目(表示采用 AH 保护)匹配时,要求 SAD 查看是否存在一个合适的 SA。如果没有,可用 IKE 动态地建立一个。如果有,就将 AH 应用到这个与之相符的数据包上,该数据包在 SPD 条目指定的那个模式中。如果它是 SPD,应用顺序就由它所涉及的协议而定。

创建一个流出 SA 时(要么人工,要么通过 IKE),将序列号计算器初始化成 0。在利用这个 SA 构建一个 AH 头之前,计算器就开始递增。这样保证了每个 AH 头中的序列号都是一个独一无二的、非零的和单向递增的数。

AH 头的其余字段都将填满恰当的值。SPI 字段分配的值是取自 SA 的 SPI;下一个头字段分配的是跟在 AH 头之后的数据类型值;而载荷长度分配的则是 AH 协议头长度减 2 的值(32 位为单位);身份认证数据字段设成 0。和 ESP 不一样,AH 将安全保护扩展到外部 IP 头的原有的或预计有的字段。因此,将完整性检查值(ICV)之前的不定字段调成零是必要的。对没有包含在认证 ICV(因此,不在保护之列)中的 IPv4 头来说,它的不定字段是 Type of service(服务类型)、Flags(旗标)、Fragment offset(分段偏移)、Time to live(存活时间)和 Header checksum(头校验和)。

对 IPv4 选项或 IPv6 扩展头来说,如果它们是固定的或预定的,都会包含在 ICV 计算之中。否则,必须在计算 ICV 之前,把它们调成零。

根据认证器的要求,可能需要进行适当的填充。对有些 MAC 来说,比如 DES-CBC MAC,要求在其上面应用 MAC 的数据必须是算法的块尺寸的倍数。在这种情况下,就必须进行填充,以便正确地使用 MAC。这个填充项是隐式添加的。它必须一概为零,其大小不包括在载荷长度中,并且不随数据包一起传送。

对 IPv4 来说,AH 头必须是 32 个字节的一倍,IPv6 则是 64 个字节的一倍。如果 MAC 的输出和这项要求不符,就必须添加 AH 头。对填充项的值没有什么别的要求,但必须把它包括在 ICV 计算中,而载荷长度中必须反映出填充项的大小。如果强制实施身份认证程序正确对齐了,在用 HMAC-MD5-96 或 HMAC-SHA-1-96 时,就不再需要填充项。

通过把密钥从 SA 和整个 IP 包(包括 AH 头)传到特定的算法(它被认作 SA 中的“身份认证程序”)这一方式,对 ICV 进行计算。由于不定字段已设成零,它们就不会包括在 ICV 计算中。接下来,ICV 值被复制到 AH 的“身份认证数据”字段中,IP 头中的不定字段就可根据 IP 处理的不同得以填充。

现在,AH 处理结束,AH 保护的 IP 包可以输出了。根据包的大小,在进行传送之前,

可将它分段，或在两个 IPSec 同级之间的传送过程中，由路由器进行分段。

2. 进入分组的处理

如果一个受安全保护的包在被收到之前，分成了几段，就要求在 AH 输入处理之前，对这些分段进行重新组合。

处理 IPSec 包的第一件事情是：根据目的 IP 地址、安全协议（AH）和 SPI，接收方确定适当的 SA。如果没有找到合适的 SA，这个包就会被丢弃。

找到 SA 之后，进行序列号检查。抗重播检查会决定这个包是新收的还是以前收到的。如果检查失败，这个包就会被丢弃。

现在必须检查完整性检查值（ICV）了。对整个数据包应用认证器算法，并将获得的摘要同保存下来的 ICV 值进行比较。如相符，IP 包就通过了身份认证；如不相符，便丢弃该包。

10.4 IKE

IPSec 使用密钥交换（Internet Key Exchange，IKE）协议实现安全协议的自动安全参数协商，可协商的安全参数包括数据加密及鉴别算法、加密及鉴别的密钥、通信的保护模式（传输或隧道模式）、密钥的生存期等，这些安全参数的总体称之为安全关联。IPSec 协议簇使用 IKE 密钥交换协议来进行密钥及其他安全参数的协商。IKE 就是 IPSec 规定的一种用于动态管理和维护 SA 的协议。用 IPSec 保护一个 IP 数据流之前，必须先建立一个 SA。SA 可以手工或动态创建。当用户数量不多，而且密钥的更新频率不高时，可以选择使用手工建立的方式。但当用户较多，网络规模较大时，就应该选择自动方式。

10.4.1 IKE 概述

1. IKE 介绍

IKE 协议的基础由以下三部分组成。

- ISAKMP 协议：由美国国家安全局（NSA）的研究人员开发，是一种密钥交换框架，独立于具体的密钥交换协议。它定义了消息交换的体系结构，包括两个 IPSec 对等体间分组结构和状态转换。
- OAKLEY 协议：由亚利桑那大学的一名安全专家 Hilarie Orman 开发的一种协议，它提出了基于模式的机制在两个 IPSec 对等体之间达成相同加密密钥。
- SKEME 协议：描述了一种通用的密钥交换技术，由加密专家 Hugo Krawczyk 设计。这种技术提供了基于公钥的身份认证和快速密钥刷新。

IKE 协议实际上是结合了 ISAKMP 协议、OAKLEY 协议和 SKEME 协议三个协议各自的优点，从而形成了一个新的混合协议。它主要利用了 ISAKMP 协议中定义的“两个阶段”思想、OAKLEY 协议中定义的“模式”思想以及 SKEME 协议的“共享和更新密钥”技术。它包括两个交换阶段，定义了四种交换模式，允许使用四种认证方法。

ESP 和 AH 用来对 IP 报文进行封装、加/解密、认证以达到保护 IP 报文的目的，而 IKE

和 ISAKMP/Oakley/SKEME 则是通信双方用来协商封装形式、加/解密算法及其密钥、密钥的生命期、认证算法的。

IKE 的用途(如图 10-9 所示)就是在 IPSec 通信双方之间,建立起共享安全参数及密钥,亦即建立"安全关联"关系。IKE 还可用于协商虚拟专用网(VPN)和远程用户(其 IP 地址不需要事先知道)访问安全主机或网络,支持客户端协商。

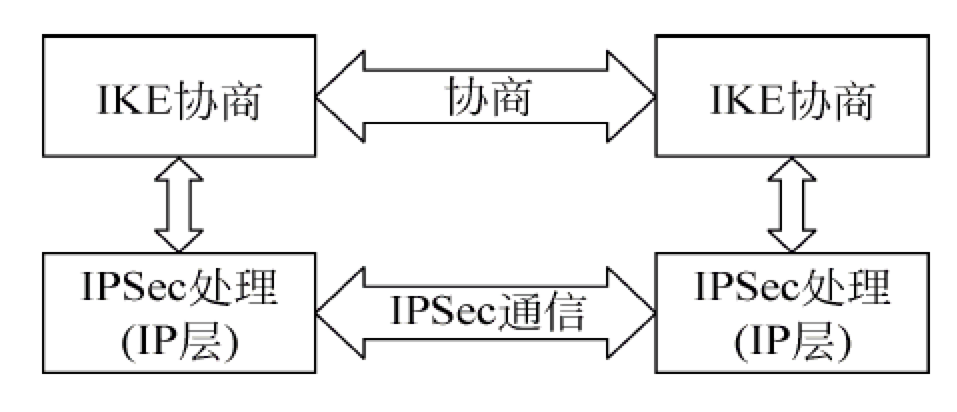


图 10-9　IKE 的用途

IKE 是一种常规用途的安全交换协议,可用于策略的磋商,以及认证加密材料的建立,适用于多方面的需求——如 SNMPv3、OSPFv2 等。

IKE 采用的规范是在"解释域"中制订的。针对 IPSec 存在着一个名为 RFC2407 的解释域(The Internet IP Security Domain of Interpretation for ISAKMP),它定义了 IKE 具体如何与 IPSec SA 进行协商。如果其他协议要用到 IKE,每种协议都要定义各自的 DOI。为正确实施 IKE,须遵守三份文档的规定,它们分别是:基本 ISAKMP 规范(RFC2408)、IPSec 解释域(RFC2407)、IKE 规范本身(RFC2409)。

IKE 主要完成两个作用:安全关联的集中化管理,减少连接时间、密钥的生成和管理。

2. IKE 涉及的密码算法

1) D-H 交换

IKE 协议利用了 Diffie-Hellman 交换算法完成通信双方密钥的分发,D-H 算法的安全性源于在有限域上计算离散对数比计算指数更加困难的事实,它能够在一个公开的不安全的信道上,为交换的双方建立起一个安全的共享秘密的通道。其算法 $1<g<n$,它们不必是秘密的,可以在公共信道上传输。

① A 选取一个大的随机数 x,并计算 $X=g^x \bmod n$,然后将 X 传送给 B。

② B 选取一个大的随机数 y,并计算 $Y=g^y \bmod n$,然后将 Y 传送给 A。

③ A 计算 $k=Y^x \bmod n=g^{xy} \bmod n$。

④ B 计算 $k_1= X^y \bmod n=g^{xy} \bmod n$。

其中 n 和 g 都是事先协商好的素数,g 和 n 可以公开,现在 $k=k_1=g^{xy} \bmod n$。将 k(即 k_1)作为 A 和 B 共享的密钥。由于窃听方只可能知道 n、g、X 和 Y,而根据 n、g、X、Y 不能恢复出 x 和 y,因而无从推断出 A 和 B 的共享密钥 k(即 k_1)。

2) HMAC

HMAC 是一种密钥化的散列算法,可由现有的任何散列算法加以运用,从而使之更加健壮。建立在使用密钥的消息的散列算法基础上的定义如下:

$$\mathrm{HMAC}(K, M)=H(K \oplus \mathrm{opad}, H(K \oplus \mathrm{ipad}, M))$$

其中:ipad 是值为 0x36 的 64bit 数组;opad 是值为 0x5c 的数 64bit 组。IPSec 中进行的所有消息验证都是采用 HMAC。常用的算法有 HMAC_md5 或 HMAC_sha。

3) 完美向前保密(PFS)

完美向前保护是指某一密钥泄露不会影响到其他密钥所保护的信息的安全性。通过一次 DH 交换产生的对称密钥,在使用一段时间后,应当丢弃,再进行新一轮 DH 交换,产生新的共享密钥,两个密钥间完全没有相关性。如果攻击者破解一个对称密钥,也无法推算出与

当前密钥无关的新密钥,因此他只能访问此密钥保护下的数据,但不能得到别的密钥保护下的数据。通常把这种采用短暂的一次性密钥的系统称为“完美向前保密(PFS)”。

3. 使用符号的介绍

HDR:ISAKMP 头,它的交换类型就是交换的模式。HDR* 表示加密 ISAKMP 头后的载荷。

SA:带有一个或多个建议载荷的安全关联载荷。

KE:密钥交换载荷。

ID_x:标识载荷。x 为 ii 时表示 ISAKMP 的发起者,x 为 ir 时表示 ISAKMP 的响应者。

HASH:散列载荷。

SIG:签名载荷。

CERF:证书载荷。

N_x:x 的 nonce 载荷。x 为 i 时代表 ISAKMP 的发起者,x 为 r 时代表 ISAKMP 的响应者。

CKY-I:ISAKMP 头中的发起者 Cookie。

CKY-R:ISAKMP 头中的响应者 Cookie。

g^x:发起者的 DH 公开值。

g^y:响应者的 DH 公开值。

prf(key,msg):使用密钥 key 和输入消息 msg 的为随机数函数,如 HMAC。

SKEYID:由仅有通信双方知道的秘密密钥信息生成的密钥串。

SKEYID_d:用来为后续 IPSec SA 生成数据加密密钥。

SKEYID_a:IKE 用来认证它的消息的密钥信息。

SKEYID_e:IKE 用来保护它的新消息的保密性的密钥信息。

4. 主要载荷说明

(1) HDR:HDR 是 ISAKMP 的通用头,每一个 IKE 消息都以它开始,其格式如图 10-10 所示。

发起者 Cookie				
响应者 Cookie				
下一个载荷	主版本	次版本	交换类型	标志 Cookie
消息 ID				
长度				

图 10-10 HDR 格式

其中,“发起者 Cookie”和“响应者 Cookie”是一对公有信息(如源/目的 IP 地址、源/目的端口等)和一些本地信息的散列值,可以帮助通信双方确定消息是否来自对方;“标志”字段主要用于加密同步;“下一个载荷”说明消息中的第一个载荷;“消息 ID”表示相同的 IKE SA 保护下的不同协议的 SA,所以在 IKE SA 建立的过程中这一字段的值为 0;“长度”字段表示 HDR 头和紧接在 HDR 之后的所有载荷的总长度。

(2) SA 载荷：携带发起者建议的一系列安全参数，供响应者选择。其格式主要部分如图 10-11 所示。

下一个载荷	保留	载荷长度
DOI		
情形		
…		

图 10-11　SA 载荷格式

其中，"下一个载荷"表示紧接的载荷类型，尽管 SA 载荷后一定跟着一个或多个建议载荷，而建议载荷后又跟着一个或多个变换载荷，但是"下一个载荷"指向的是变换载荷后紧接的载荷；"载荷长度"为整个 SA 载荷的长度(包括建议载荷和变换载荷)；"保留"字段必须为 0；"DOI"制定协商所基于的 DOI；"情形"表明协商发生时的情形。

(3) 建议载荷：依据 SA 载荷"情形"字段的值，发起者在发 SA 载荷的建议载荷部分给出建议接收者使用的安全协议。其格式如图 10-12 所示。

下一个载荷		保留		载荷长度	
建议数	协议 ID		SPI 大小	变换数	
变长的 SPI					

图 10-12　建议载荷格式

其中，"下一个载荷"指向下一个建议载荷，或者为 0(后面没有载荷时)；"保留"字段必须为 0；"载荷长度"为整个建议载荷的长度(包括该建议载荷包含的变换载荷)；有时可能会建议选择使用多个安全协议，因此每一个建议载荷必须有一个"建议数"，如果这些安全协议之间是逻辑"与"的关系，则对应的建议载荷的"建议数"必须相同；如果安全协议之间是逻辑"或"的关系，则对应的建议载荷的"建议数"单调增长；"协议 ID"表示本载荷建议使用哪一个安全协议，阶段 1 只能选择 ISAKMP 协议，阶段 2 可以选择 AH、ESP、COMP 或它们的逻辑组合；"SPI"大小表示"变长的 SPI"字段值的长度，阶段 1 没有使用到 SPI，所以这个字段的值为 0；"变换数"表示包含在该建议载荷中的变换载荷的个数。

(4) 变换载荷：用于建立 SA 的协商中为一个制定协议提供不同的安全机制。格式如图 10-13 所示。

下一个载荷	保留	载荷长度
变换数	变换ID	保留2
SA属性		

图 10-13　变换载荷格式

其中，"下一个载荷"或者指向下一个变换载荷，或者为 0(后面没有变换载荷时)；"保留"字段必须为 0；"载荷长度"为本载荷的字节数；因为可能有多个变换，因此每一个变换必须对应一个"变换数"，并且这些"变换数"是单调增长的；"变换 ID"说明建议协议的变换标识符；"保留 2"也必须为 0；"SA 属性"包含由"变换 ID"字段指定的变换的 SA 属性，该属性用来在 IKE 通信双方之间传递具体的参数，如加密算法、散列算法、操作模式等。

(5) 密钥交换载荷 KE：用于传输密钥交换数据。发起者和响应者都在本地计算 DH 公钥，然后将该公钥编码入 KE 载荷。

(6) nonce 载荷 N_i：包含一个随机数，保护交换数据免受重放攻击。

(7) 标识载荷 ID_X：用于确认 SA 协商发起者的身份，响应者用身份信息来确定应用于 SA 的安全策略。

5. 交换与交换模式

由于 IKE 以 ISAKMP 为框架,所以它使用了两个交换阶段,第一个阶段用于建立 IKE SA,第二个阶段利用已建立的 IKE SA 为 IPSec 协商具体的一个或多个安全关联,即建立 IPSec SA。同时 IKE 定义了交换模式,即主模式(main mode)、积极模式(aggressive mode)又称野蛮模式、快速模式(quick mode)以及新群模式(new group mode)。

在不同的交换阶段可以采用的交换模式不同,具体作以下介绍。

1) 阶段 1 交换

在阶段 1,主要任务是创建一个 IKE SA,为阶段 2 交换提供安全保护。

阶段 1 交换包括主模式交换和积极模式交换:主模式将 SA 的建立和端对端身份认证以及密钥协商相结合,使得这种模式能抵抗中间人攻击;积极模式简化了协商过程,但抵抗攻击的能力较差,也不提供身份保护。它们均在其他任何交换之前完成,用于建立一个 IKE SA 及验证过的密钥。其中主模式的特点包括身份保护和对 ISAKMP 协商能力的完全利用。其主要包括:协商保护套件、执行 Diffie-Hellman 交换、认证 Diffie-Hellman 交换及认证 IKE SA。

与 IPSec SA 不同的是,IKE SA 是一种双向的关联:IKE 是一个请求-响应协议,一方是发起者(initiator),另一方是响应者(responder)。一点建立了 IKE SA,将同时对流入和流出业务进行保护。IKE SA 提供了各种各样的参数,它们是由通信实体双方协商制定的。这些参数成为一个“保护套件”——包括散列算法、认证算法、Diffie-Hellman 组、加密算法等。

2) 阶段 2 交换

在阶段 2,主要任务是在 IKE SA 的保护下,创建 IPSec SA。

一个阶段 1 的 SA 可以用于为 IPSec 建立一个或多个 SA。这样,通过协商适当的 IPSec SA,建立了通信对等方(如安全网关)之间的安全关联。由于阶段 2 交换受阶段 1 协商好的 IKE SA 的保护,所以在阶段 2 中使用快速模式。在快速模式下交换的载荷都是加密的。

新群模式用于为 Diffie-Hellman 密钥交换协商一个新的群。新群模式是在 ISAKMP 阶段 1 交换中建立的 SA 的保护之下进行的,同快速模式一样,在新群模式下交换的载荷也都是加密的。

在两个交换阶段中,阶段 2 交换是在阶段 1 建立的 IKE SA 的保护下进行的,而阶段 1 交换是在没有任何安全保护的情况下进行的,所以 IKE 充分使用四种认证方法。这四种认证方法分别是:基于数字签名的认证、基于公钥加密的认证、基于修订的公钥加密的认证和基于预共享密钥的认证。IKE 规定在上述两个阶段、四种模式下,阶段 1 主模式和阶段 2 快速模式必须实现。下面就主要介绍四种认证下必须实现的模式。

10.4.2 协商过程

1. 阶段 1 协商

阶段 1 交换可以使用主模式和积极模式,其中规定主模式必须实现。无论使用哪一种模式交换,通信双方建立共享秘密的唯一方式是采用 Diffie-Hellman 交换。Diffie-Hellman 算法的优点是可以通过即使是不安全的途径协商共享秘密,其安全性源于在有限域上离散

对数的计算比指数计算更为困难；它的缺点是无法抵御中间人攻击，因为参与密钥协商过程的通信双方没有办法验证与它们会话的另一方是谁。为了弥补 Diffie-Hellman 密钥交换协议的不足，在 IKE 阶段 1 交换中增加了对 Diffie-Hellman 交换的共享秘密进行认证的步骤，同时对所建立的 IKE SA 本身进行认证。

在阶段 1 交换中，无论使用哪种模式，都可以采用数字签名、公钥加密、修订的公钥加密和预共享密钥这四种认证方法，但交换消息的载荷组成因认证方法的不同而有所差异。

不同认证模式下密钥信息 SYEKID 的计算方式也是不同的。

基于数字签名的认证：$SKEYID_{sign}=prf(N_i|N_r, K_{DH}(I,R))$；

基于公钥加密的认证：$SKEYID_{pd}=prf(hash(N_i|N_r), K_{DH}(I,R))$；

基于预共享密钥认证：$SKEYID_{pd}=prf(K_{pre}(I,R), N_i|N_r)$。

主模式或积极模式都最终生成经认证的密钥，并根据具体的安全策略来保护此后的通信数据：

SKEYID_d=prf(SKYID,g xymodn|CKY_I|CKY_R|0)

SKEYID_a=prf(SKYID,SKYID_d|gxymodn|CKY_I|CKY_R|1)

SKEYID_e=prf(SKYID,SKYID_a|g xymodn|CKY_I|CKY_R|2)

其中以 SKEYID 为基础衍生出后续密钥，SKEYID_d 为用于衍生的加密材料，SKEYID_a 用来提供消息完整性和对数据源进行身份认证，SKEYID_e 用于对 IKE 消息加密。

为了保护交换数据的完整性，发起者和响应者分别生成认证码 HASH_I 和 HASH_R。当使用数字签名认证时，HASH_I 和 HASH_R 需经过生成方的签名并被对方验证；而使用其他认证方式时，HASH_I 和 HASH_R 直接可以认证被交换的数据。

$HASH_I=prf(SKEYID,\ g^x modn|g^y modn|CKY_I|CKY_R|SA_{i_b}|ID_{ii})$

$HASH_R=prf(SKEYID,\ g^y modn|g^x modn|CKY_R|CKY_I|SA_{i_b}|ID_{ir})$

下面主要对协商过程进行介绍。

1）使用预共享密钥的阶段 1 交换

预共享密钥验证方式是最简单的一种认证方法。发起方和响应者必须事先利用非密码技术协商好一个或多个不同的预共享密钥，并为具体的某次交换标识出所使用的某个密钥。

（1）预共享密钥认证主模式

主模式下使用预共享密钥验证方式的消息交换过程如图 10-14 所示。

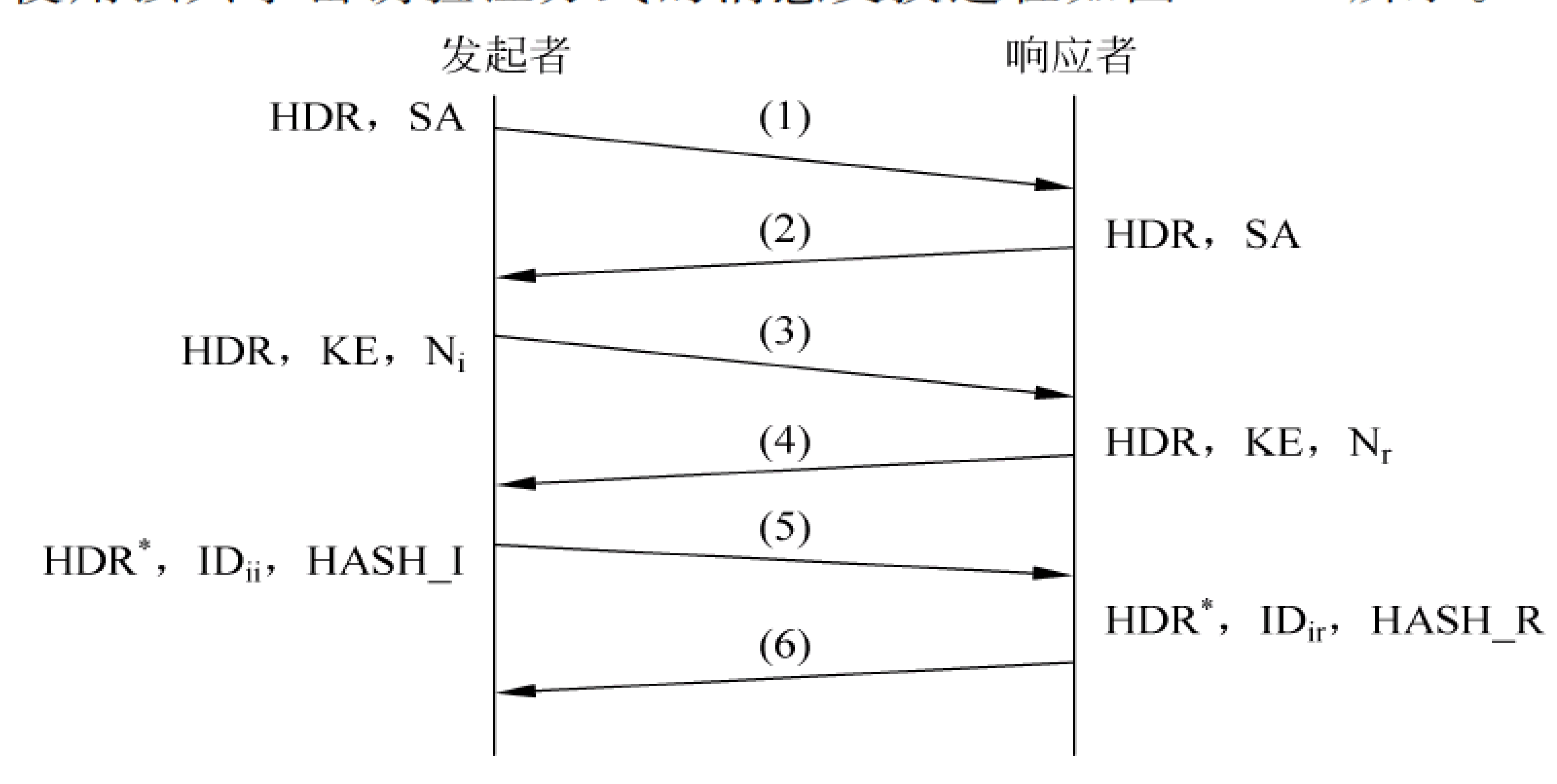


图 10-14　预共享密钥认证的主模式

主模式的交换过程中，前两个消息用于协商 SA 的属性，报文以明文传输，没有经过认证。发起者可以建立含有一个或多个提议的 SA_i，响应者不能修改任何建议的属性。发起者在第一条消息 ISAKMP 头中的发起者 Cookie 字段，插入自己的 Cookie，此时响应者 Cookie 字段为 0，在第二条消息中，响应者才填入一个自己的 Cookie。在以后的交换中它们始终存在而且不能改变。这些 Cookie 是一种长 8 个字节的伪随机数，对于一个特定的交换来说，它们与 SPI 标识 IPSec SA 一样，由<Cky_I，Cky_R>唯一地标识 IKE SA，它可以防止重放攻击。Cookie 由源 IP 地址、端口号、目的地址、时间和只有生成者才知道的一个秘密生成。具体的 Cookie 的生成方法如下：

CKY_I=MD5(secret_i，源 IP|目的 IP|源 UDP 端口号|目的 UDP 端口号|时间 i)

CKY_R=MD5(secret_r，源 IP|目的 IP|源 UDP 端口号|目的 UDP 端口号|时间 r|CKY_I)

中间两个消息分别用于交换 D_H 的公共值和当前时间(Nonce)即 N_i 和 N_r，建立一个主密钥(SKEYID)。Nonce 是用于产生新的密钥材料并阻止通过重放攻击产生的虚假的安全联盟。

最后两个消息用来传送相互认证所需要的信息。ID 用来认证身份，HASH_I 和 HASH_ R 散列摘要用来证明消息 1 到 4 的完整性。两条消息都是先用 SKYID_e 加密，然后用 SKEYID_a 认证。

使用预共享密钥验证方式的主模式消息交换预共享密钥的缺陷在于：由于 SKEYID 的生成依赖于参与方的共享密钥，所以如果不知道最后两条消息的接收方使用的是哪个预共享密钥时，对方将不能解密两条消息的 ID 信息。一种解决办法是根据报文中的源 IP 地址来查找相应的预共享密钥，这给一个不使用静态 IP 地址的环境下使用预共享密钥认证带来很大的局限性。解决方法之一就是使用基于公钥的认证方法，但如果必须使用预共享密钥认证的方式，则可以采用积极模式交换。

(2) 预共享密钥认证积极模式

预共享密钥认证积极模式的用途与主模式交换相同，都是为了建立一个验证的安全关联和密钥，随后可用 IKE 为其他安全协议建立安全关联。主要差别在于，积极模式只需要用到主模式一半的消息。

使用预共享密钥认证积极模式交换如图 10-15 所示。第一条消息包含一个"保护套件"列表、DH 公共值、Nonce 以及一些身份认证的数据；第二条消息是一个响应，对响应方进行身份认证，并回应一个选择好的"保护套件"、DH 公共值、Nonce 以及身份认证的数据，此时已经交换了所有用于 IKE SA 加密密钥的信息，因此最后一条消息可以加密，但不是必需的；最后一条消息用于认证发起者的身份。

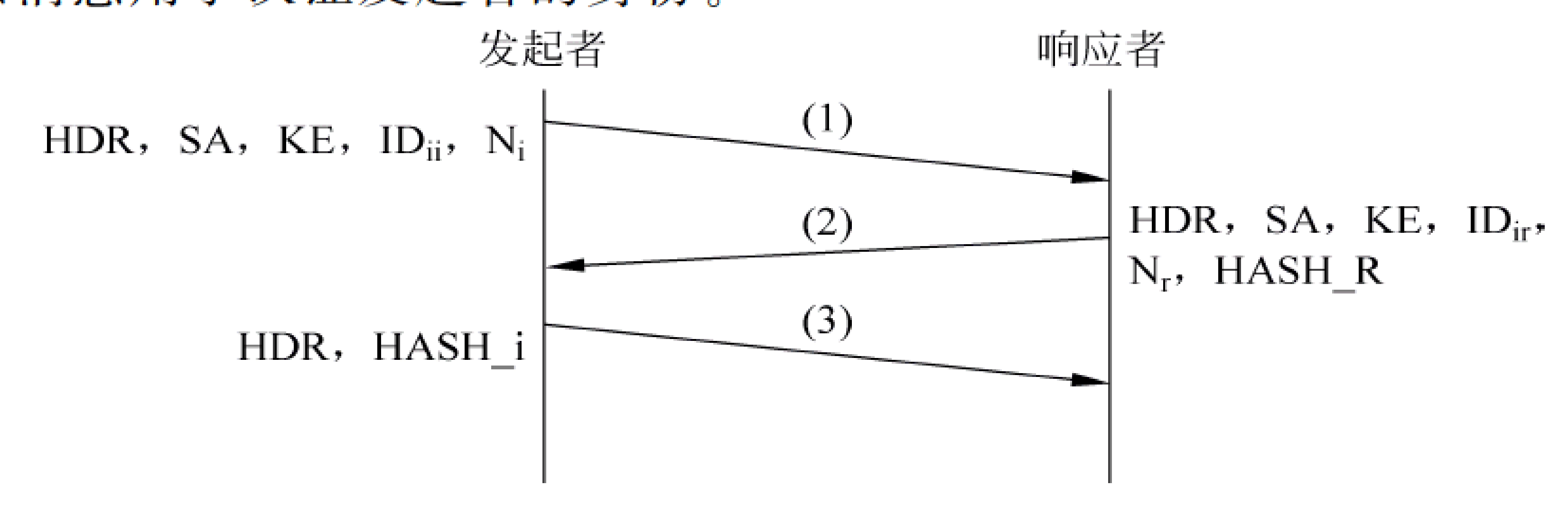


图 10-15 预共享密钥认证积极模式

预共享密钥认证积极模式交换的局限性：

(1) 积极模式不能体现 IKE 丰富的协商功能，积极模式对消息量的限制也导致了对该模式的协商能力的限制。

(2) 不提供对通信双方的身份保护，因为所有的认证信息一次就发送完毕了。

(3) 由于发起者必须在第一条消息中提供它的 DH 公共值以及它的 Nonce，因此在不同的“保护套件”中无法提供不同的 DH 公共值。

(4) 积极模式限制了加密算法和散列算法的使用。

2) 使用数字签名的阶段 1 交换

数字签名是签字人首先用单向哈希函数求得原报文的报文摘要，再用 RSA 算法中的私钥加密报文摘要，就生成了“数字签名”，接收方用签字人的公钥解密“数字签名”，得到签字人发来的报文摘要，然后再计算收到报文的报文摘要，将两个报文摘要进行比较，若相同，则验证了报文及签名是真实完整的。

使用数字签名的主模式下交换由六个消息构成，其过程如图 10-16 所示。

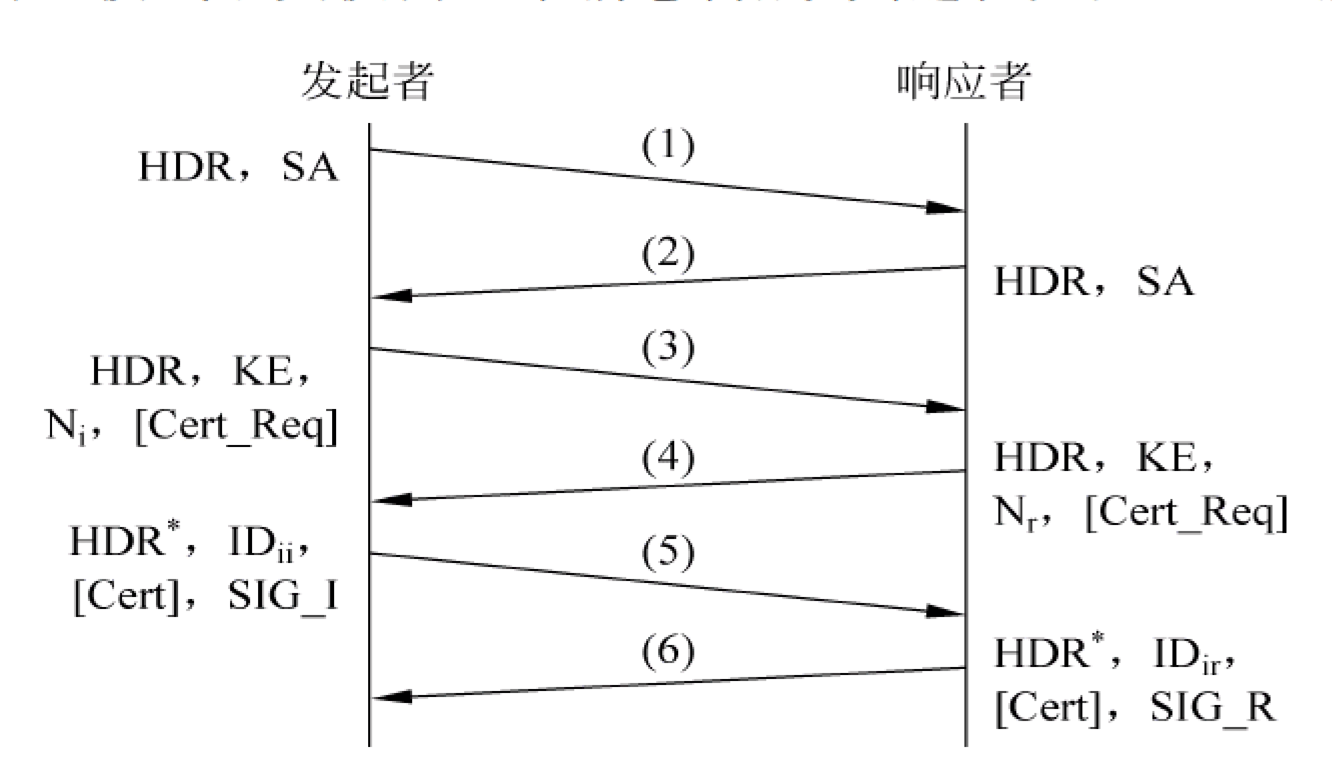


图 10-16　数字签名认证的主模式

前两个消息用于建立发起者和响应者之间达成一系列安全参数的共识，包括加密算法、散列算法、身份认证方式、OAKLEY 群等。这些安全参数主要用于对身份信息和阶段 2 交换的保护，其中，加密算法将用于对后续的 IKE 协商的加密保护；散列算法用于对交换消息进行完整性保护；身份认证方式表明对对方的身份的鉴别方式；OAKLEY 群选择 Diffie-Hellman 交换的基群。

中间两个消息完成 Diffie-Hellman 交换，为通信双方生成一个共享的秘密密钥，通信双方都将依据此秘密，进行独立的密钥衍生。KE 是密钥交换载荷；N_i/N_r 是发起方/响应方产生的瞬时载荷。

后两个消息主要用于对已交换的 IKE 消息进行一致性检查，以及对端进行身份验证。HDR^* 表示随后的整个消息都使用在之前已协商好的密钥和加密算法。ID_{ii}/ID_{ir} 表示发起者/响应者的身份鉴别载荷；SIG_I/SIG_R 分别表示发起者和响应者的数字签名载荷，对 HASH_I 和 HASH_R 进行数字签名；[Cert]表示证书载荷可选。

3) 使用公钥加密的阶段 1 交换

使用公钥加密的身份验证交换，其交换的辅助信息是加密的矫时载荷 nonce 和实体的身份信息 ID。交换的双方只有能重新推导出 HASH 值才算通过验证。

使用公钥加密的主模式消息交换过程如图 10-17 所示。

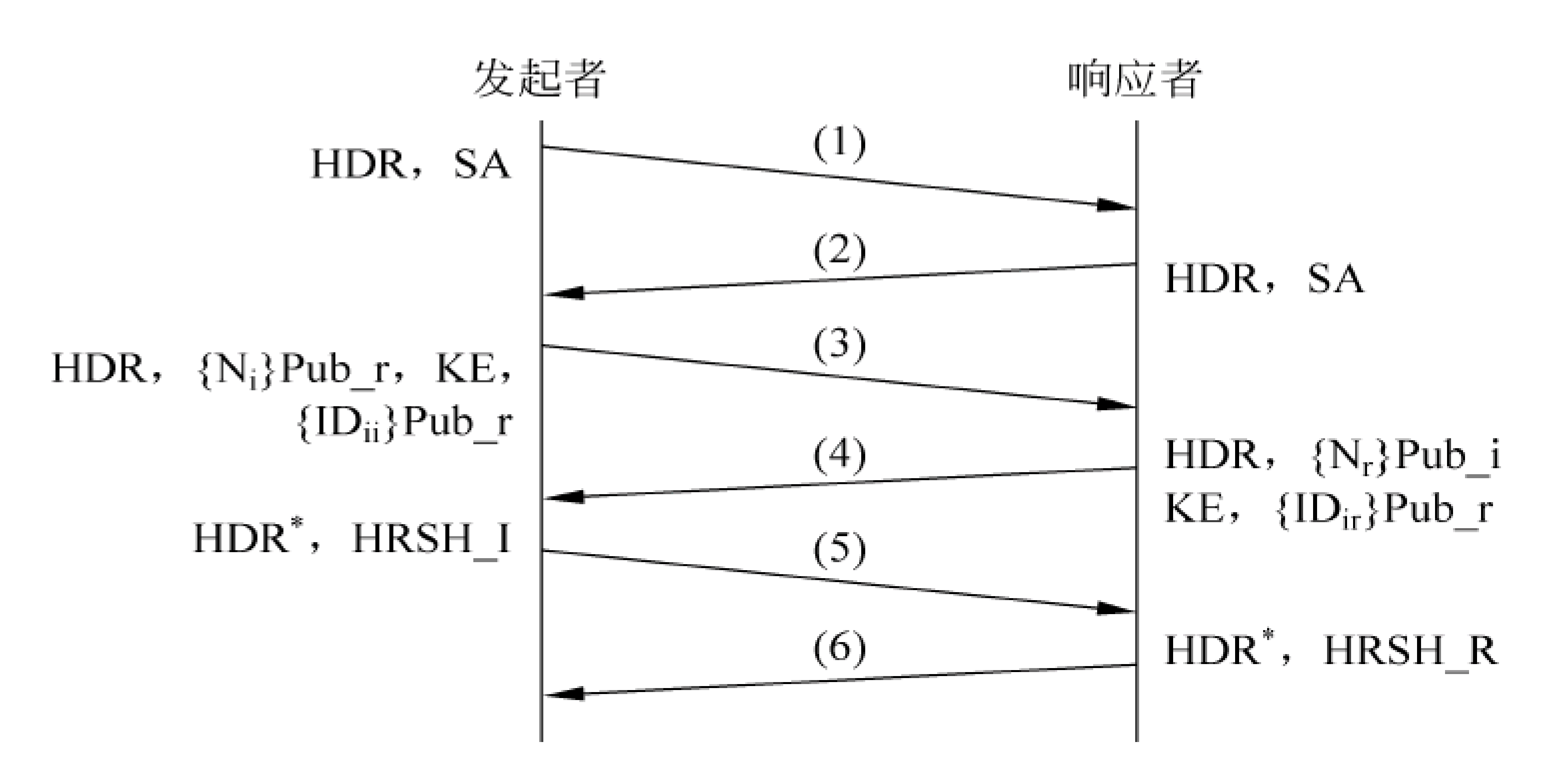


图 10-17 公钥加密认证的主模式

Pub_i/Pub_r 是使用发起方/响应方的公钥进行加密的载荷，分别是身份鉴别载荷和矫时载荷，因此在第三和第四个消息处理完成后即通过了身份鉴别。而第五和第六两个消息用于验证双方消息交换的一致性。

上述公钥加密认证方法的缺点是：在协商的过程中总共需要进行四次耗时的加密运算。所以，对公钥加密认证方法进行了改进。

4）使用改进的公钥加密的阶段 1 交换

改进的公钥加密认证主模式只需要一般的公钥运算，而且发起者可以向响应者出示证书。然而发起者仍然需要使用 IKE 之外的方法来获得响应者的证书。主模式下改进的公钥加密验证方式的消息交换过程如图 10-18 所示。其中，{}ke_x 表示用密钥“ke_x”进行对称密钥加密，Ke_i＝prf(N_i，CKY-I)，Ke_r＝prf(N_r，CKY-R)。

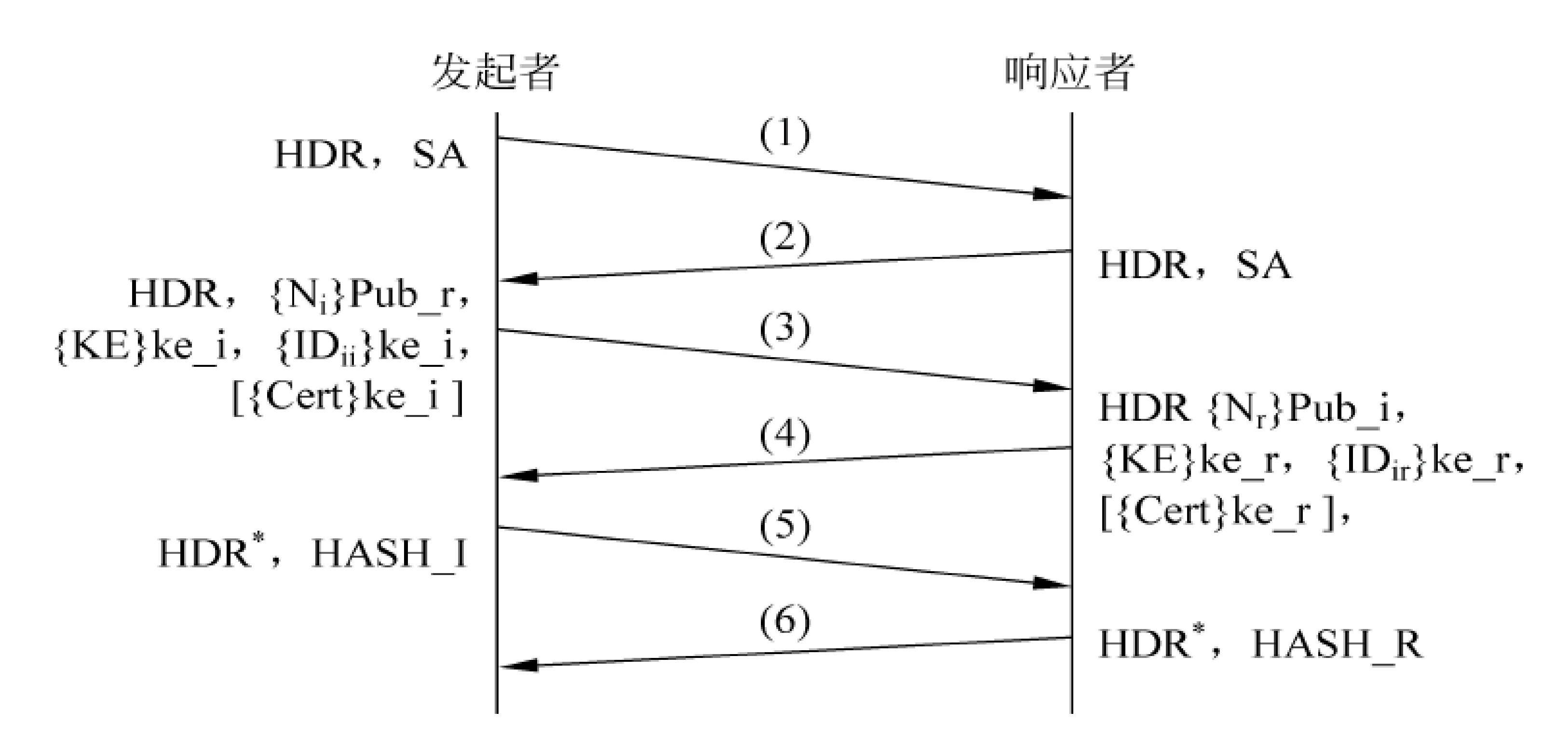


图 10-18 改进的公钥加密认证的主模式

5）其他积极模式交换

数字签名认证积极模式、公钥加密认证积极模式以及修订的公钥加密认证积极模式的交换步骤也如图 10-18 所示，区别仅在于后两个消息中的验证载荷的表现形式不同：对于基于预共享密钥的认证以及基于公开密钥的认证来说，是一个散列载荷；而对于基于数字签名的认证来说，是一个签名载荷，但是不管哪种积极模式都存在前面所阐述的局限性。

积极模式(又称野蛮模式)适用于首先考虑带宽的场合，或者一方对另一方的策略有比较全面的了解，那么利用这种交换方式就可以更快地建立 IKE SA，而没有必要利用主模式协商全部功能。

2．阶段 2 协商

在完成了前述的 IKE 阶段 1 交换即建立起 ISAKMP SA 之后，可以利用它来进行阶段 2 交换即为 IPSec 协议生成相应的 IPSec SA。

1）快速模式

阶段 2 交换定义了快速交换模式，它使用 ISAKMP SA 来保护通信。快速模式中，消息由 ISAKMP 头、HASH 载荷、一个或多个 SA 载荷以及其他载荷组成。一个 ISAKMP SA 可用于保护多个同时发生的快速模式协商，由于这些协商都要使用在阶段 1 交换生成的 cookie 对，为了能够区分这些同时进行的协商，必须为每个协商指定唯一标识，具体做法是在 IKE 报文的 ISAKMP 报头中加入消息标识 M_{ID}。

快速交换模式仅通过交换三条消息来完成具体协议的安全关联的协商。发起方根据本地的策略要求，通过 SA 载荷提议一种或多种保护数据通信的安全协议（如 ESP 或 AH），并给出相应的变换（即安全协议的安全参数）。响应方根据本地策略的要求，从提议的一种或多种安全协议（如 ESP 或 AH）中选择一种，并从未选中的安全协议给出的一种或多种保护套间中选取一种，作为选择后的安全关联（SA）来保护数据通信。快速模式中包含的瞬时载荷用来产生新的密钥材料。每个快速模式可以交换一个可选的 KE 载荷来允许另外的 DH 交换。是否包含 KE 载荷将决定本次 IKE 协商能否提供“完美前向保密（Perfect Forward Secrecy，PFS）”服务。

快速模式交换过程如图 10-19 所示。

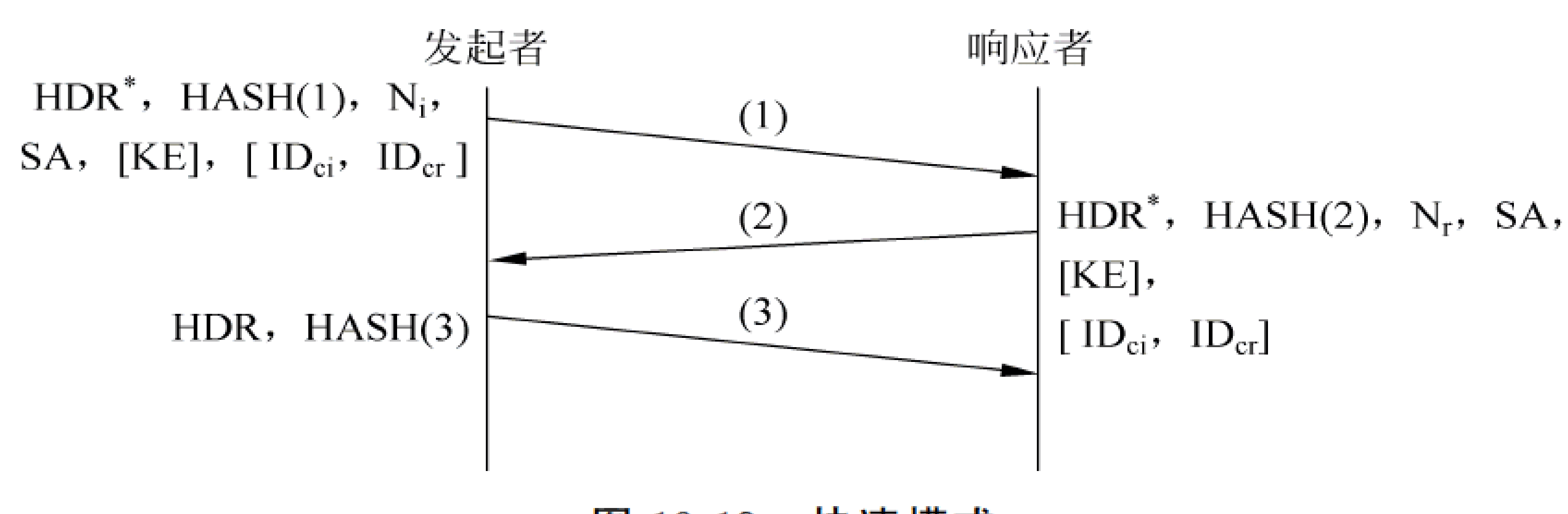


图 10-19　快速模式

IKE SA 保护快速模式的方法有：用 SKEYID_e 加密所有交换的消息，提供机密性服务；对所有的消息进行认证，不仅提供数据完整型保护，还可以对数据源的身份进行验证，即在接到消息后，可以验证它是否来自认证通过的实体，以及那条消息在传输过程中有没有被篡改。

快速模式不是一个完整的交换，它需要阶段 1 形成的 SKEYID_a，如果阶段 1 正常完成，则可以保证 SKEYID_a 的机密性。

3 个消息中都使用了散列载荷，用以保护交换消息的完整性。其计算方式为

$$\text{HASH}(1)=\text{prf}(\text{SKEYID_a},\ M_{ID}\,|\,SA\,|\,N_i\,|\,[KE]\,|\,[\ ID_{ci},ID_{cr}])$$

$$\text{HASH}(2)=\text{prf}(\text{SKEYID_a},\ M_{ID}\,|\,N_{i-b}\,|\,SA\,|\,N_r\,|\,[KE]\,|\,[\ ID_{ci},\ ID_{cr}])$$

$$\text{HASH}(3)=\text{prf}(\text{SKEYID_a},\ 0\,|\,M_{ID}\,|\,N_{i-b}\,|\,N_{r-b})$$

快速模式需要从 SKEYID_d 状态中衍生出用于 IPSec SA 的密钥。由于所有 IPSec 密钥都衍生自相同的源，所以相互之间是有关联的。如果攻击者得到 SKEYID_d，那么衍生出

来的 IPSec SA 密钥都是不安全的。为此,IKE 协议提供了完美向前保密(PFS)。所谓PFS,指即使攻破了第一阶段的密钥交换(即攻破了 SKEYID 等衍生密钥),也只能阅读受该 SA 保护的信息,却不能阅读收 IPSec SA 保护的信息。

消息(1)、(2)中如果不包含 KE 载荷,则 IKE 协议规定新的密钥材料生成方法为

$$KEYMAT = prf(SKEYID_d, protocol \mid SPI \mid N_{i-b} \mid N_{r-b})$$

显然,如果攻破了 SKEYID_d 等,则 KEYMAT 很容易被第三方计算出来,此时不能实现 PFS。

消息(1)、(2)中如果包含 KE 载荷,则 IKE 协议规定新的密钥材料生成方法为

$$KEYMAT = prf(SKEYID_d, g^{xy} \mid protocol \mid SPI \mid N_{i-b} \mid N_{r-b})$$

其中:g^{xy}表示快速模式下 DH 交换生成的暂时共享密钥。此时,即使第三方攻破了SKEYID_d,也只能阅读受保护的 g^{xy},而根据离散对数的难解性,第三方仍不可能知道KEYMAT,这就实现了 PFS。

在第一、第二两个消息中,分别包含了两个身份载荷。这两个载荷与阶段 1 的身份载荷的作用是不一样的。它们不用于鉴别目的,而在于为将建立的 IPSec SA 协商构造选择符。这个选择符规定了该 IPSec SA 将保护什么样的通信。这两个消息中还包含了两个Nonce 载荷,这两个载荷的目的在于给对方一个存活的证明,表明自身确实是通信的对方,而不是冒充者。因为冒充者不能从第一个消息中解密出 N_i,因此不能由自身的消息计算出合法的 HASH(2)。

IPSec SA 的建立过程受到 IKE SA 的机密性、完整性保护。而且,通过 HDR 中的Cookie 字段以及 Nonce 载荷,使建立过程在一定程度上能抵抗重放攻击和拒绝服务攻击。

2) 新群模式

新群模式用于为 Diffie-Hellman 密钥交换协商一个新的群。新群模式是在 ISAKMP阶段 1 交换的 SA 的保护之下进行的,如图 10-20 所示。

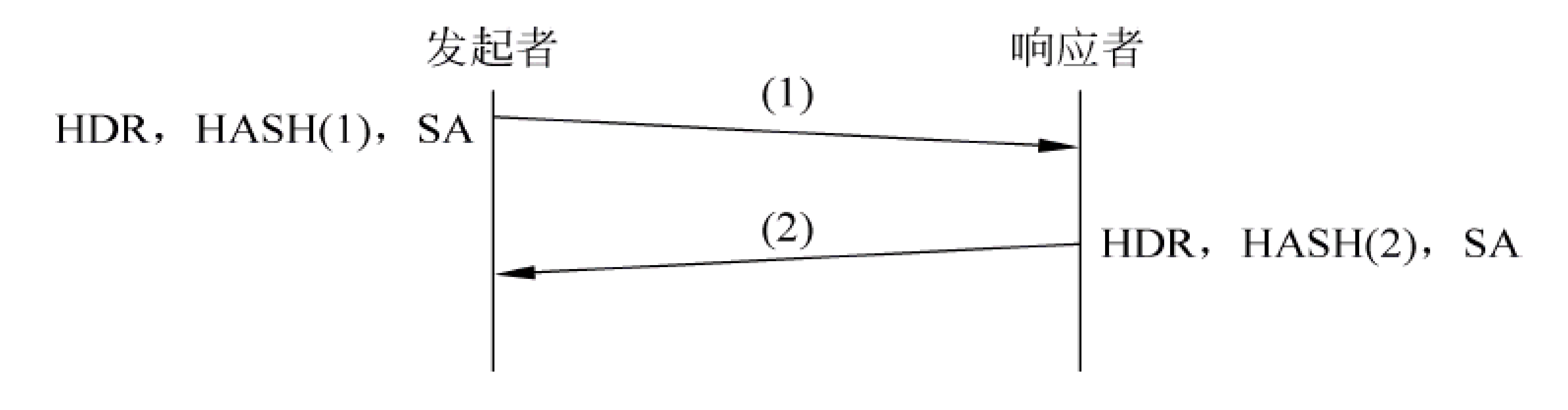


图 10-20 新群模式交换过程

新群模式用于为 Diffie-Hellman 密钥交换协商一个新的群。新群模式是在 ISAKMP阶段 1 交换的 SA 的保护之下进行的。在第一条消息中,发起方送出一个 SA 载荷,其中包含了新群的特征(例如模指数运算的指数和底数),如果响应者能够接受这个群,便在第二条消息中用完全一样的信息作出回应,否则拒绝该提议。在新群交换中,HASH(1)和 HASH(2)的计算为

$$HASH(1) = prf(SKEYSTR_a, MsgID \mid SA)$$

$$HASH(2) = prf(SKEYSTR_a, MsgID \mid SA)$$

习　题　10

1. IPSec 可以防范哪些攻击？

2. 传输模式和隧道模式的区别是什么？

3. 什么是安全关联？

4. 在 IPSec 中，IPSec 策略由安全策略数据库加以维护。对一个 SPD 条目来说，它可以定义哪几种行为？

5. IPSec 提供哪些功能？

6. 在 AH 中，是不是对 IP 报头中的所有字段计算 MAC，为什么？

7. 简述一下 IKE 协议的工作原理。

第 11 章　Web 安全

目前随着 Internet 尤其是 WWW 的日益普及，人们对网络的依赖越来越强，网络已经成为人们生活中不可缺少的一部分，但是随着网络应用的深入，当人们享受着互联网带来的方便快捷时，各种类型的用户加入到网络中，出于政治、经济和商业的目的，网站成为攻击目标。Web 安全问题已经成为 Internet 发展中必须面对的重要问题。

11.1　Web 的基本概念与相关技术

Web 又称 World Wide Web（环球信息网，中文名是“万维网”），是由文档和文档之间的链接构成的庞大信息网。Web 设计中引入了三个重要概念，即统一资源定位符（Uniform Resource Locator，URL）、超文本传输协议（Hypertext Transfer Protocol，HTTP）和超文本标记语言（Hypertext Markup Language，HTML）。Web 的基本结构是采用开放式的客户机/服务器结构（Client/Server），分成服务器端、客户端、传输规程三个部分。服务器规定传输设定、信息传输格式和服务器本身的开放式结构；客户机称浏览器，用于向服务器发送资源索取请求，并将收到的信息进行解密和显示；传输规程是 Web 浏览器与服务器之间进行通信传输的规范。

1. URL

为了在广阔的 Internet 世界中找到所需的信息，组成 Web 的网页必须拥有唯一的地址。网页的唯一标识符叫做统一资源定位符（URL）。URL 是 Web 中用来寻找资源地址的方法，表示 Web 服务器的地址以及获取服务所用的传输协议。服务器地址可以是本机磁盘，也可以是 Internet 上的站点。URL 的一般形式如下：

＜URL 的访问方式＞://＜主机＞:＜端口＞/＜路径＞

2. HTTP 协议

HTTP 协议是分布式 Web 应用的核心技术，在 TCP/IP 协议栈中属于应用层。它定义 Web 浏览器向 Web 服务器发送索取 Web 页面请求格式以及 Web 页面在 Internet 上的传输方式。其工作过程是：Web 服务器在 80 端口等候 Web 浏览器的请求；Web 浏览器通过三次握手与服务器建立 TCP/IP 连接，然后 Web 浏览器通过类似-GET/dailynews.html 形式的简单命令向服务器发送索取页面的请求，服务器则以相应的文件为内容响应 Web 浏览器的请求。

3. HTML

Web 服务的特点在于高度的集成性，它能够实现不同类型的信息（如文本、图像、声音、动画和视频等）和服务（如 News、FTP、Telnet 以及 Mail 等）的无缝链接，特别适用于广域网

中信息的组织、检索与显示，如 Internet 上的信息、广告、新闻、电子商务、电子政务和展示信息等，从而使得 Web 技术越来越受到人们的关注。这些特点决定了 Web 的内容必须能够以适当的形式来组织和安排，使得它在各种平台的 Web 浏览器上能够得到正确的解释，并具有层次丰富的界面，如文本、图形图像和链接等应该具有不同的诠释和显示。

HTML 的出现解决了页面作者定制网页总体轮廓的问题，用文本语言的方式实现了 Web 内容和存储上的统一。HTML 几乎为所有常见的 Web 浏览器所支持。Web 浏览器在得到 Web 页面之后，根据 HTML 语言的标记来决定页面的层次结构和显示格式，并且可以通过 URL 来实现 Web 页面的链接和跳转。HTML 中可以包括层叠样式表（Cascading Style Sheets，CSS），CSS 属于一种样式设计模板，它能够帮助用户控制 HTML 元素的呈现方式和轮廓，将 HTML 的内容制作和样式设计分开。

4. Web 服务器

Internet 上众多的 Web 服务器汇集了大量的信息，Web 服务器的作用就是管理这些文档，处理用户发来的各种请求，并将满足用户要求的信息返回给用户。

本质上来说，Web 服务器是驻留在服务器上的一个程序，通过 Web 浏览器与用户交互操作，为用户提供感兴趣的信息。

5. Web 浏览器

Web 浏览器是阅读 Web 上信息的客户端软件。Web 浏览器在网络上与 Web 服务器打交道，从服务器上下载和获取文件。

Web 浏览器有多种，它们都可以浏览 Web 上的内容，只不过所支持的协议标准以及功能特性各有异同。绝大部分的浏览器都运用了图形用户界面。目前常用的有：Netscape Navigator、Netscape Communicator、Microsoft Internet Explorer、Opera、Mosaic 以及 Lynx 等。Netscape 的浏览器几乎可以在所有的平台上运行，而 Microsoft Internet Explorer 则是占据了 Web 浏览器的主要市场。浏览器是最经常使用到的客户端程序。

6. 公共网关接口

公共网关接口（Common Gateway Interface，CGI）是 Web 服务器向基于 CGI 的程序传递数据的标准。它提供一个计算机程序同 HTTP 协议或者 WWW 服务的接口，也就是人机交互接口。CGI 程序（在 CGI 控制下运行的程序，通常称 CGI 程序）可以使用任何语言编写，但它通常使用 C/C++或 Perl 语言编写。CGI 脚本的典型用法是从 Web 页面上获取、处理信息再向用户提供其他所需的信息。

CGI 的主要功能是在 WWW 环境下，从客户浏览器传递一些信息给 WWW 服务器，再由 WWW 服务器去启动所指定的程序代码（CGI 程序）来完成相应的工作。CGI 是信息服务器主机对外信息服务的标准接口，为了向客户端提供动态信息而制定。使用 CGI 可以像网关一样，在服务器端和客户端之间建立一个桥梁，通过执行客户端的输入指令，产生并传回客户端所需要的信息。

CGI 是服务器的扩展，以扩充 Web 服务器能力，编写的服务器扩展使得 Web 站点更生动有趣。同时 CGI 是非会话式的，它不能在处理来自 Web 浏览器的数据过程中停止并返回 Web 浏览器请求更正数据，CGI 只允许数据输入。CGI 处理数据，生成输出，然后终止。

在 HTML 文件中，表单与 CGI 程序配合使用，共同完成信息交流的目的。一般过

程是：

① 用户用 Web 浏览器提交表单登录；

② Web 浏览器发送登录请求到 Web 服务器；

③ Web 服务器分析 Web 浏览器送来的数据包，确认是 CGI 请求，于是通过 CGI 将表单数据按照一定格式发送给相应的 CGI 应用程序；

④ CGI 应用程序对数据处理、验证，将动态生成的页面发送给 Web 服务器；

⑤ Web 服务器把 CGI 应用程序发来的页面发送给请求登录的 Web 浏览器；

⑥ Web 浏览器接受、解释、显示页面。

Web 浏览器向 Web 服务器提交表单数据通常有两种方式：一是 Post 方式，Web 服务器通过标准输入方式把数据转交 CGI 应用程序；CGI 应用程序数据处理完毕后，将结果输出到标准输出，即可以为 Web 服务器所接受；二是 Get 方式，在 UNIX 类的系统中，Web 服务器通过环境变量将数据转交 CGI 应用程序。

11.2 Web 攻击

11.2.1 拒绝服务攻击

拒绝服务攻击 DoS，是一种破坏网络服务的技术方式，其目标是让被攻击的计算机停止提供服务或资源访问。

发起 DoS 攻击的攻击者往往利用系统的缺陷，运行特定编写的程序，这些程序或者耗尽网络带宽，或者使目标系统或服务崩溃，其最终结果就是使合法用户的请求得不到满足，进而会使一个依赖于计算机或网络服务的企业不能正常运转。

2000 年 2 月 9 日，著名的搜索引擎 Yahoo、电子商务网站 Amazon、eBay 和 CNN 等遭到强烈的分布式拒绝服务攻击，服务器连续几十个小时无法正常工作，造成的损失高达 12 亿美元。

拒绝服务攻击的基本目标是阻止受入侵的计算机提供正常服务或资源访问，使得合法者无法正常使用网络服务。计算机和网络运行时都需要一定的资源，例如网络带宽、存储器、磁盘空间和数据结构等。拒绝服务攻击最常见的就是入侵者通过产生大量的导向受害网络的数据包，消耗该网络的所有可用带宽，典型的攻击包是 ICMP Echo 包。例如在 Smurf 中，入侵者从远端节点向某网络的广播地址发送 ICMP Echo 请求包，网络上的所有节点都会响应这一请求，从而产生大量垃圾数据包，最终使得网络拥塞或瘫痪。

除了网络带宽，入侵者还可以消耗其他的网络资源。如针对网络连接，阻止受害的主机或网络同其他网络进行通信。TCP SYN Flood 入侵就是这样一种入侵方式，入侵者与受害节点建立连接，但不最终完成。由于受害节点需要保持并等待完成这些半开的连接，结果导致合法的连接因为缺乏数据结构资源而无法正常建立连接。在这种入侵中，入侵者消耗的是核心的数据结构（主机资源），而不是网络带宽。这意味着入侵者可以通过一个慢的网络入侵一个高速的网络上的主机，这是一种典型的“非对称攻击”。另外还有很多系统中保持信息的数据结构，如进程描述符、进程表项和进程时隙等，都是有限的。入侵者可以编写一

个脚本程序，通过不断地自我复制消耗这些资源或 CPU 时间。

此外，入侵者通过破坏或改变配置信息，如改变网络路由信息、改变 Windows NT 的注册表信息等，也可以阻止计算机或网络的使用，甚至通过破坏计算机或网络中的物理组件导致拒绝服务。

拒绝服务的攻击方式有很多种，一般是利用系统的漏洞、协议的漏洞、服务的漏洞对计算机发动攻击，使计算机无法正常对外服务。除了利用系统漏洞、协议漏洞来进行，拒绝服务攻击也可以利用合理的服务请求来占用过多的服务资源，致使服务过载，无法响应其他用户的合法请求。这些服务资源包括网络带宽、系统文件空间、连接的进程数量等。常见拒绝服务攻击的方法主要有以下几种：

(1) 制造高流量的无用数据，造成网络拥塞，使受害主机无法正常和外界通信；

(2) 利用受害主机提供的服务或协议上的缺陷，反复而高速地发出特定的服务请求，使受害主机无法及时处理所有正常请求；

(3) 利用受害主机所提供的服务中处理数据上的缺陷，反复发送畸形数据引发服务程序错误，大量占有系统资源，导致目标主机无法处理而拒绝服务。

11.2.2　Web 欺骗

在访问网上银行时，根据所访问的银行 Web 页面，从该行的账户中提取或存入一定数量的存款。因为相信所访问的 Web 页面就是所需要的银行的 Web 页面。无论是页面的外观、URL 地址，还是其他一些相关内容，都感到非常熟悉，所以感到没有理由不相信。但是，其实很可能已经被不法分子所欺骗。

2003 年 12 月 5 日，一个网址是 www.hkhsbc.com 的假冒网站出现，这与香港汇丰银行网站的网址“www.hkbc.com.hk”非常相似。假网站上有一个用户登入的按钮，与香港汇丰银行的用户登入按钮一模一样，假如用户不慎在假网站输入用户名称及密码，款项便会被不法之徒攫取。

2003 年 12 月 10 日，中银香港宣布发现了一个假冒站点 www.bocfund.com。经查证，它是一家总部设在深圳名为“中银互惠理财”的公司。该公司在其网页利用“中银国际集团”的名义替投资者开设账户，买卖香港及美国证券，还公开采用了中银“BOC”的简称。

Web 欺骗是互联网上一种常见的欺骗攻击方式，攻击者伪造某个 www 站点，使得对该伪造站点的访问都进入到攻击者的 Web 服务器，并经过攻击者计算机的过滤作用，从而达到攻击者能监视被攻击者的任何活动以获取有用信息(用户的账号、口令等)的目的，攻击者可以伪造数据以某个用户的身份发送给服务器，或者以服务器的身份发送伪造的数据给用户。

Web 欺骗是一种电子信息欺骗，攻击者创建表面上可信的、伪造的 Web 页面。伪造的 Web 看起来十分逼真，它拥有相同的网页和链接。然而，攻击者控制着伪造的 Web 站点，被攻击者的浏览器和所访问的 Web 之间的所有网络信息完全被攻击者所截获，其工作原理就像一个过滤器，攻击者可以对用户的通信进行监视和篡改。

Web 欺骗的手段和方法主要有以下几种。

1) URL 重写

Web 攻击实现的关键是攻击者将 Web 页放置在用户和其他的 Web 之间。利用 URL

地址改写把 Web 上的 URL 进行重写，使得链接指向攻击者的服务器。URL 重写不用将网络路由配置成通过一个代理服务器改变所有网络通信的路径，也不需要配置浏览器。它是通过在 Web 页面中改变 URL，URL 重写代理服务器运用特定的策略映射“真实 URL”，使用户访问许可数据库请求的浏览器自动转向返回到代理服务器的请求。

假设攻击者的主机为 www. hide. com，通过在所有链接前增加 http：//www. hide. com，攻击者可以改写 Web 页中的所有 URL 地址。这样经过改写后，原来正常的 URL www. cumt. edu. cn 就变成了 http：//www. hide. com http：//www. cumt. edu. cn。当用户单击改写过的 www. cumt. edu. cn 时，进入的是 http：//www. hide. com，然后由 http：//www. hide. com 向 http：//www. cumt. edu. cn 发出请求并获得真正的内容，再由 http：//www. hide. com 返回给用户浏览器。

由于浏览器一般均设有地址栏和状态栏，当浏览器与某个站点链接时，可以在地址栏和状态栏中获得链接中的 Web 站点地址及其相关的传输信息，用户由此可以发现问题。但是由于浏览器的显示是由 Web 编写者定义的，所以攻击者往往在 URL 地址重写的同时，加入 JavaScript 程序，因为 JavaScript 程序具有修改地址栏和状态栏内容的功能，以达到掩盖欺骗的目的。

例如，图 11-1 给出了 Web 欺骗的工作方式。

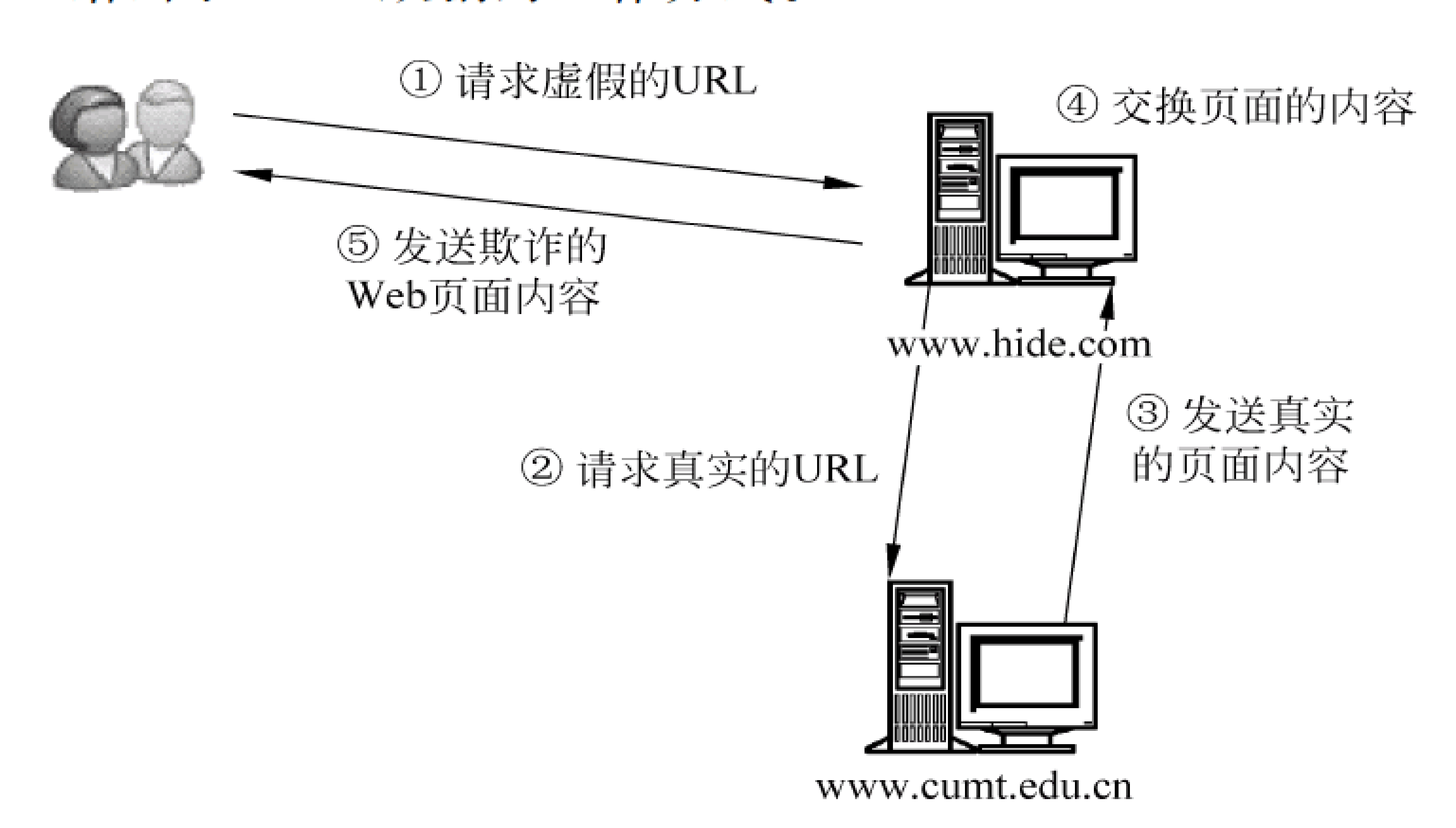


图 11-1 Web 欺骗的工作方式

2）DNS 域名重定向

通常用户对 Web 站点的访问是基于客户机/服务器模式的，其基本过程是：在浏览器地址栏输入域名→向 DNS 查询 IP 地址→利用 DNS 返回的 IP，建立 TCP/IP 连接→向服务器发送 HTTP 请求→服务器返回客户机浏览器请求的内容→关闭连接。

从上述过程可知，客户机首先要获取域名对应的 IP 地址后，才能向对应的服务器发送连接请求。客户机从默认的域名服务器中获取域名对应的 IP 地址，而如果客户机的默认的 DNS 服务器没有域名对应的 IP 地址，由域名服务器在网络中进行递归查询，在其他域名服务器上获取地址信息。

如果攻击者通过某种方法（攻破 DNS 服务器，DNS 欺骗）把用户要访问的目标机器的域名对应的 IP 改成攻击者所控制的机器，这样所有外界对该域名的请求将转向攻击者的机器。这时攻击者可以转发所有的请求到目标机器，让目标机器进行处理，再把处理结果返回

到发出请求的客户机。这样，所有外界进入目标域名机器的数据都在攻击者的监视之下，攻击者可以任意窃听甚至修改数据流里的数据，收集到大量的信息。

3）站点欺骗技术

攻击者假冒一个合法的银行站点，假冒的站点看起来和合法的站点一模一样，至少极其相似，使用户很难通过察看网页内容判断访问站点的真伪，这个在技术上不难实现。

4）Phishing 技术

自 2004 年以来，Phishing“钓鱼式攻击”、“网络钓鱼”已经给国内外用户造成了很大的损失，或信用卡账号被盗，或银行账户上的存款不翼而飞。网络钓鱼（Phishing）一词，是“Fishing”和“Phone”的综合体，由于黑客始祖起初是以电话作案，所以用“Ph”来取代“F”，创造了 Phishing，Phishing 发音与 Fishing 相同。中文有一个很好的典故“姜太公钓鱼，愿者上钩”；网络钓鱼者垂钓于网络中，诱饵千百种，其行为目的很明确，就是以利为目的，盗取银行的存款或其他机密信息，例如，通过以下几种方式“钓鱼”。

（1）发送电子邮件，以虚假信息引诱用户中圈套

诈骗分子以垃圾邮件的形式大量发送欺诈性邮件，这些邮件多以中奖、顾问、对账等内容引诱用户在邮件中填入金融账号和密码，或是以各种紧迫的理由要求收件人登录某网页提交用户名、密码、身份证号、信用卡号等信息，继而盗窃用户资金。

（2）建立假冒网上银行、网上证券网站，骗取用户账号密码实施盗窃

犯罪分子建立起域名和网页内容都与真正网上银行系统、网上证券交易平台极为相似的网站，引诱用户输入账号密码等信息，进而通过真正的网上银行、网上证券系统或者伪造银行储蓄卡、证券交易卡盗窃资金；还有的利用跨站脚本，即利用合法网站服务器程序上的漏洞，在站点的某些网页中插入恶意 Html 代码，屏蔽一些可以用来辨别网站真假的重要信息，利用 cookies 窃取用户信息。

（3）利用木马和黑客技术等手段窃取用户信息实施盗窃

木马制作者通过发送邮件或在网站中隐藏木马等方式大肆传播木马程序，当感染木马的用户进行网上交易时，木马程序即以键盘记录的方式获取用户账号和密码，并发送给指定的邮箱，用户资金将受到严重威胁。

11.2.3 DNS 欺骗

DNS 欺骗是一种比较复杂的攻击技术，因为它使用起来比 IP 欺骗要简单，所以比较常见。最近一个利用 DNS 欺骗进行攻击的案例，是全球著名网络安全销售商 RSA Security 网站的 DNS 中的域名曾被修改，当用户尝试打开 RSA Security 的主页时，发现其主页已变成了其他的内容。DNS 欺骗攻击是指 DNS 服务器接受并使用一些来自未授权主机的错误信息，导致当其他用户向 DNS 查询地址时，DNS 服务器返回错误的结构。

域名管理系统（Domain Name System，DNS）是域名解析服务器，为方便记忆，互联网上的主机都采用以名称（如 www.cumt.edu.cn）的形式作为用户访问该主机的网址。然而实际上，网络上的主机之间的通信不是用网址来标识的，而是用 IP 地址。因此当用户要访问某一站点时，用户程序首先需要知道该站点的 IP 地址，DNS 服务器的功能就是负责将用户输入的网址解析为 IP 地址。

假设客户端程序 A 在运行，A 希望访问一个 Web 服务器 B，知道 B 的网址，但又不知道

B 的 IP 地址,这时就需要通过 DNS 查询来获得这个地址(如图 11-2 所示)。

(1) 首先 A 向 DNS 服务器 Z 发送 DNS 查询请求。

(2) DNS 服务器处理这个查询请求,首先搜索本地记录,如果查找到关于 B 的 DNS 记录,将 B 的 IP 地址信息返回给 A;如果 Z 在本地没有 B 的 DNS 记录,则还要发送请求到其他的 DNS 服务器,然后将其他服务器返回的 DNS 查询结果返回给 A。

(3) 主机 A 利用 DNS 返回的地址和主机 B 通信。

从这个过程,可以看到 DNS 有两个工作特性:①为提高效率,DNS 系统会对所有已经经过查询的结果进行缓存(Cache);②DNS 对自身无法解析的域名,会自动向其他 DNS 服务器查询。正是由于以上特点,使得入侵者能进行 DNS 缓存中毒(DNS Cache Poisoning)攻击。

图 11-3 为 DNS 的查询和响应报文的格式,报文由 12 字节的首部和 4 个长度可变的字段组成。标识字段通常称为“标识 ID”,由客户程序设置,并由服务器程序返回结果。客户程序通过它来确定响应与查询是否匹配。

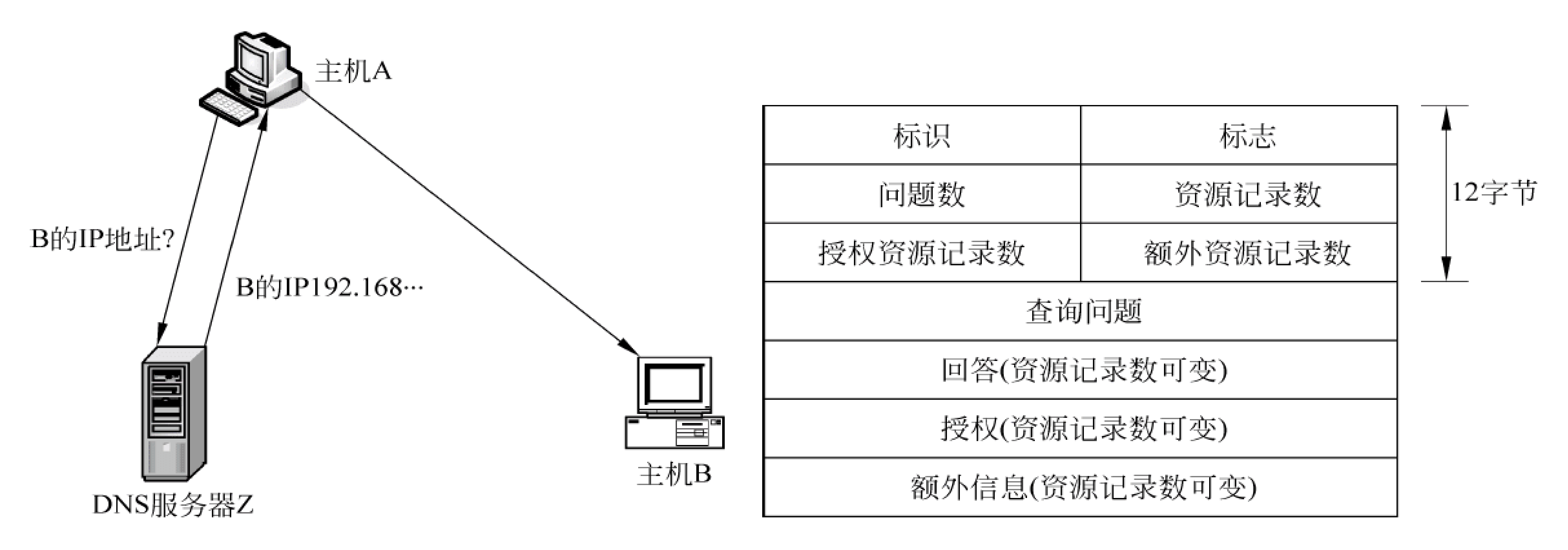


图 11-2　DNS 的工作过程　　**图 11-3　DNS 查询包的结构**

DNS 欺骗的难点在于 DNS 查询包中的“标识 ID”。当 DNS 服务器向其他的服务器发送查询包时,在返回的 DNS 应答包中,只有“标识 ID”与其发送的查询包中的“标识 ID”吻合时,才能被 DNS 服务器所接受。

通常 DNS 服务器在构造查询包时,查询包的“标识 ID”每次自动加 1,因此攻击者可以先向将要欺骗的 DNS 服务器发一个查询包,并用 Sniffer 监听其 ID 值,然后攻击者再发送一个查询包,紧接着发送一系列构造好的应答包,这些应答包内的“标识 ID”为预测的一定范围内的值,例如预测在“标识 ID”的范围为(ID + 1) ~ (ID + 100)之间。如果预测的某个值正确,那么 DNS 就会把这个包当成自己查询的应答,将应答中的数据记录入缓存,而真正的应答包到来时将被 DNS 服务器丢弃。

假设,有主机 A 的 IP 为 192.168.0.3,DNS 服务器 S1 的 IP 为 192.168.0.20,Web 服务器 www.aaaa.com 的 IP 地址为 10.1.1.1。攻击者可以从主机 A 发动一次 DNS 欺骗,过程如图 11-4 所示。

① 攻击者先向 DNS 服务器 S1 提交查询主机 www.aaaa.com 的 IP 地址的请求。

② DNS 服务器 S1 中并未有主机 www.aaaa.com 的 IP 地址信息的记录, DNS 服务器

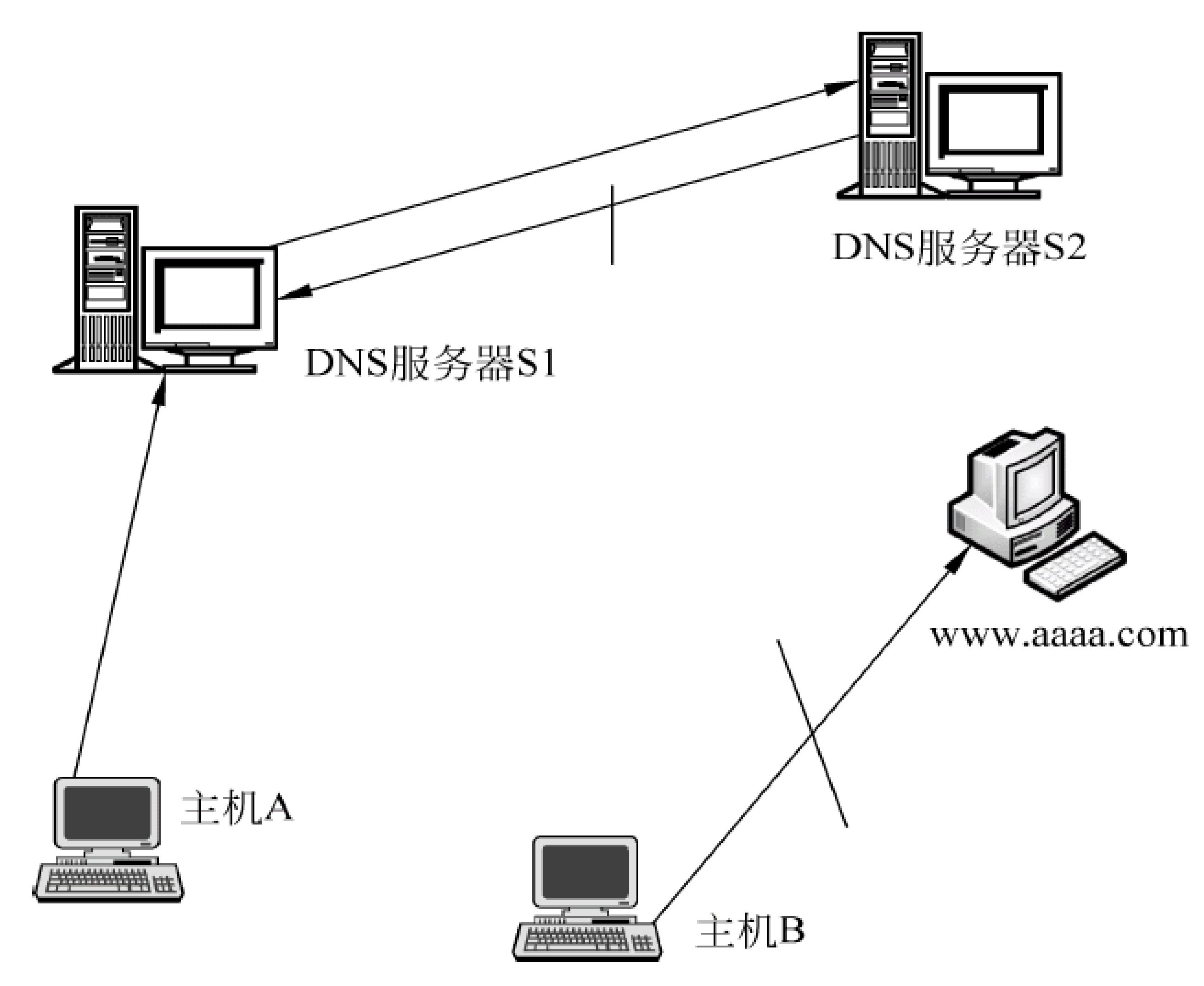


图 11-4 DNS 欺骗的过程

就向其他的 DNS 服务器(假设为 S2)递交查询请求,而此时攻击者在主机 A 上监听 DNS 服务器的通信情况,得到其 DNS 查询请求的 ID(假设为 6543)。

③ 攻击者再向 DNS 服务器 S1 提交查询请求,且立刻伪造出大量应答包响应 DNS 服务器的查询,伪造的应答包中告诉 DNS 服务器 S1 主机 www.aaaa.com 的 IP 地址是 10.0.0.0,且查询包使用的"标识 ID"的范围为 6544~6644,这样查询包中的应答信息很可能被 DNS 服务器 S1 缓存记录,而 DNS 服务器 S2 返回的信息被 S1 丢弃。

④ 当主机 B 向 DNS 服务器进行查询 www.aaaa.com 的 IP 地址时,由于 DNS 服务器 S1 中已有记录,DNS 服务器 S1 就会返回"www.aaaa.com 的 IP 地址是 10.0.0.0"的应答信息。显然造成主机 B 发送访问请求到 10.0.0.0,而不是 www.aaaa.com 的真正 IP 地址。

DNS Cache Poisoning 的限制因素主要有以下两点。

首先,攻击者不能替换缓存中已经存在的记录。比如说,如果在 98.76.54.32 这个 DNS 服务器上已经有一条 www.bbbcccc.com 的 IP 记录,那么攻击者试图替换为 www.bbbbdddd.com 将不会成功。然而,一些记录可以累加,比如 A 记录,如果在 DNS 的缓存中已经存在一条 www.bbbcccc.com 的 A 记录为 1.2.3.4,而攻击者却欺骗 DNS 服务器说 www.bbbcccc.com 的 A 记录为 4.3.2.1,那么 www.bbbcccc.com 将会有两个 A 记录,客户端查询时会随机返回其中一个。

其次,DNS 服务器存在缓存刷新时间问题,如果 www.bbbcccc.com 的 TTL 为 7200,那么 DNS 服务器仅仅会把 www.bbbcccc.com 的信息缓存 7200 秒或者说两个小时。如果攻击者放入一条 A 记录,过了默认的两天后,该记录就会被缓存清除。

通过对 DNS 服务器的缓存写入错误信息是一种攻击方式,且实现存在一定的难度,那么对客户端的 DNS Spoofing 欺骗攻击方式则容易实现得多。

前面分析可知,由于 DNS 协议在设计上的缺陷,在 DNS 报文中只使用一个序列号来进行有效性鉴别,并未提供其他的认证和保护手段,这使得攻击者可以很容易地监听到查询请求,并伪造 DNS 应答包给 DNS 客户端,从而进行 DNS 欺骗攻击。目前所有 DNS 客户端处理 DNS 应答包的方法都是简单地信任首先到达的数据包,丢弃所有后到达的,而不会对数

据包的合法性作任何的分析。这样,只要能保证欺骗包先于合法包到达就可以达到欺骗的目的,通常这是非常容易实现的。

DNS 对客户端的欺骗与 DNS Cache Poisoning 的原理相同,如图 11-5 所示,位于同一局域网的主机 A 192.168.0.2、主机 B 192.168.0.3、首选 DNS 服务器 S 192.168.10.200、远程 Web 服务器 www.aaaa.com 的 IP 地址为 10.1.1.1,则攻击者通过主机 A 发动对主机 B 的 DNS 欺骗,欺骗过程如下:

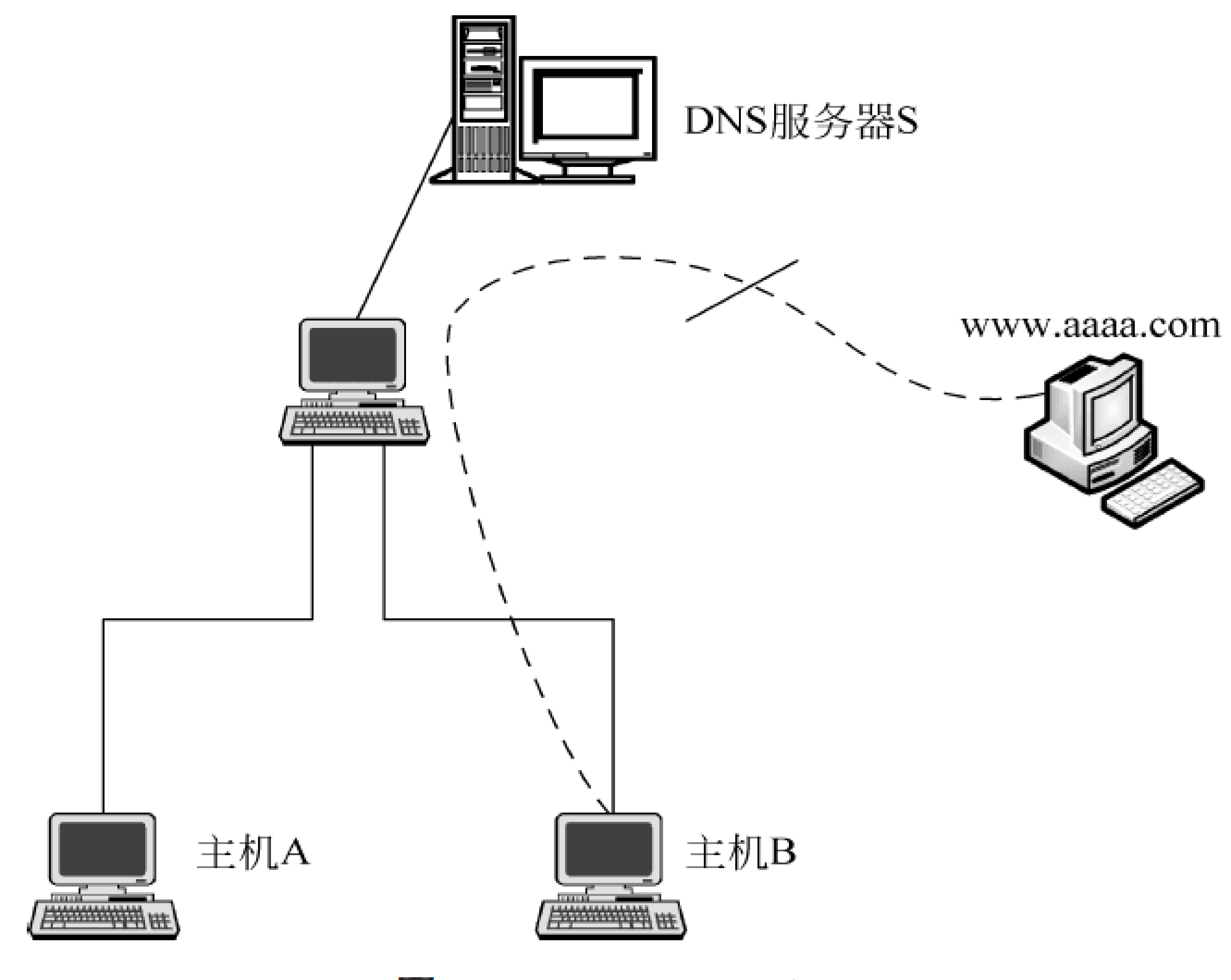


图 11-5 DNS Spoofing

(1) 主机 B 希望访问 www.aaaa.com,但不知道它的 IP 地址,则 B 的 DNS 客户端向首选 DNS 服务器 S 发送对于 www.aaaa.com 的解析请求。

(2) 攻击者在主机 A 监听到 B 发送的请求,并根据请求 ID 向请求者发送虚假应答包,告诉 B 与 www.aaaa.com 对应的 IP 地址为 123.234.0.1。

(3) 本地 DNS 服务器返回正确应答,但由于在时间上晚于攻击者的应答,结果被丢弃。

(4) 主机 B 与 123.234.0.1 通信,攻击者达到目的。

11.3 TLS 协议概述

安全传输层协议(Transport Layer Security Protocol,TLS)用于在两个通信应用程序之间提供保密性和数据完整性,它可以在不同层次提供网络安全服务。通用的解决方法是在网络层使用 IPSec,IPSec 对于最终用户和应用程序是透明的。另一个比较通用的解决方法是在 TCP 上实现安全性,在这一级,有两种实现选择,一是 SSL(Secure Sockets Layer,SSL)或 TLS 可以作为基本协议簇的一个部分提供,因此对应用程序透明,二是将 SSL 嵌入到软件中,如嵌入到 Web 浏览器与 Web 服务器中。与应用有关的安全服务也可以被嵌入到特定的应用程序中,如安全电子交易(SET)。几种安全机制如图 11-6 所示。

SSL 采用公开密钥技术。其目标是保证两个应用间通信的保密性和可靠性,可在服务器和客户机两端同时实现支持。SSL 协议要求建立在可靠的传输层协议(例如:TCP)之

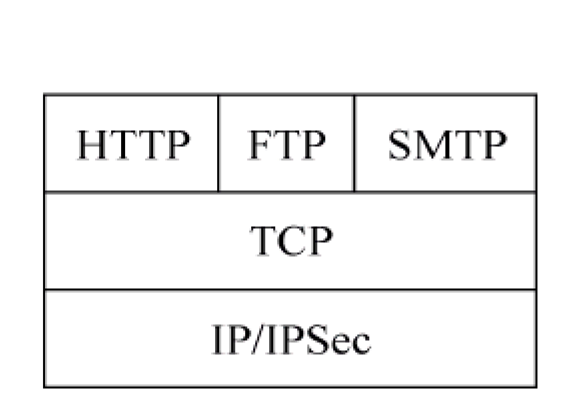


图 11-6　TCP/IP 协议栈中安全机制的位置

上。SSL 协议的优势在于它是与应用层协议独立无关的，高层的应用层协议（例如：HTTP，FTP，Telnet）能透明地建立于 SSL 协议之上。SSL 协议在应用层协议通信之前就已经完成加密算法、通信密钥的协商以及服务器认证工作。

Netscape 公司推出 Web 浏览器时提出了 SSL 安全通信协议，SSL 协议目前已成为 Internet 上保密通信的工业标准。现行 Web 浏览器普遍将 HTTP 和 SSL 相结合，来实现安全通信。如果利用 SSL 协议来访问网页，其步骤包括以下几步。

（1）用户：在浏览器的地址栏里输入 https：//www. sslserver. com。

（2）HTTP 层：将用户需求翻译成 HTTP 请求，如 GET /index. htm HTTP/1. 1，Host http：//www. sslserver. com。

（3）SSL 层：借助下层协议的信道，安全地协商出加密密钥，并用此密钥来加密 HTTP 请求。

（4）TCP 层：与 Web Server 的 443 端口建立连接，传递 SSL 处理后的数据。

接收端与上述过程相反。

1999 年，IETF 基于 SSL 3. 0 协议正式发布了 RFC 2246，并将其称为 TLS（Transport Layer Security），从技术上讲，TLS 1. 0 与 SSL 3. 0 的差别非常微小，但二者不具有互操作性。在微软 IE6 版本中，对于加密的 HTTPS 连接，IE 默认使用了 SSL 协议。用户只有通过在“高级选项”中自行设置才可使用 TLS 协议，IE7 直接启用 TLS。

自从 TLS 1. 0 成为工业标准以后，TLS 1. 0 在 Internet 上已得到长足的应用，除了如 S/HTTP、S/MIME、SSL-Telnet、SSL-SMTP、SSL-POP3 等常用的协议以外，人们在开发各种电子商务和电子政务系统时也是基于 TLS。

在 WAP 的环境下，由于手机及手持设备的处理和存储能力有限，wap 论坛（www. wapforum. org）在 TLS 的基础上做了简化，提出了 WTLS 协议（Wireless Transport Layer Security），以适应无线的特殊环境。

TLS 协议由两层组成，如图 11-7 所示。最底层是 TLS 记录层协议（TLS Record Protocol），它基于可靠的传输层协议（如 TCP/IP），用于封装各种高层协议，提供消息源认证、数据加密以及数据完整性服务。高层协议对 TLS 的会话和管理提供支持，常称为 TLS 握手协议层，主要包括 TLS 握手协议（TLS Handshake Protocol）、修改密码格式协议（Change Cipher Spec Protocol）、警告协议（Alert Protocol）、应用数据协议（Application Data Protocol）等子协议。

在 TLS 子协议中，最重要的是 TLS 握手协议。它是认证、交换协议，也对在 TLS 会话、连接的任一端的安全参数以及相应的状态信息进行协商、初始化和同步。握手协议执行完后，应用数据就根据协商好的状态参数信息通过 TLS 记录协议发送。

TLS握手协议	TLS改变密码格式协议	TLS警告协议	HTTP，FTP，…
TLS 记录协议			
TCP			
IP			

图 11-7 SSL 记录协议和 TLS 握手协议

TLS 协议的一个优点是它对于高层应用协议的透明性，高层应用数据可以使用 TLS 协议建立的加密信道透明地传输数据。同时，TLS 协议不依赖于低层的传输协议。可以建立在任何能够提供可靠连接的协议之上，例如 TCP、SPX 等。

TLS 记录协议建立在其他可靠的传输协议之上（如 TCP）。TLS 记录协议提供的连接安全性有两个基本特点。

（1）连接是保密的。在数据加密中使用了对称密码算法（如 DES，RC4）。对于每个连接，都要根据 TLS 握手协议进行密钥协商，产生一个唯一的会话密钥。TLS 记录协议也可以不加密使用。

（2）连接是可靠的。消息的传输使用了带密钥的 MAC 进行完整性检查。在计算 MAC 时使用了安全的 Hash 函数（如 SHA，MD5）。记录协议也可以不通过 MAC 来操作。

TLS 握手协议位于 TLS 记录协议以上。它使客户端和服务器在传送数据和接收数据前可以认证相互的身份，并协商加密算法和密钥，是一个建立安全信道的交互过程。TLS 握手协议提供的连接安全性具有以下三个基本特点。

（1）对等实体可以使用公钥密码算法（如 RSA，DSS）进行认证。这种认证是可选的，但是通常至少需要对一方进行认证。

（2）共享秘密的协商是安全的。即使攻击者能够发起中间人攻击，协商的秘密也不可能被窃听者获得。

（3）协商是可靠的。攻击者不能在不被发现的情况下篡改协商通信数据。

TLS 主要工作流程包括：网络连接建立；与该连接相关的加密方式和压缩方式选择；双方的身份识别；本次传输密钥的确定；加密的数据传输；网络连接的关闭。

TLS 协议定义了两个通信主体：客户和服务器。其中客户是协议的发起者。

TLS 协议中有两个重要的概念：会话和连接。

连接：是指提供一种合适服务的传输，一个 TLS 连接是瞬时的，每个 TLS 连接与一个 TLS 会话关联。连接状态包括了如下元素：服务器随机数和客户随机数、服务器写 MAC secret、客户写 MAC secret、服务器写密钥、客户写密钥、初始向量、序列号。

会话：是指客户和服务器之间的关联。会话由握手协议创建，它定义了一套安全加密参数，这套加密参数可以被多个连接共享。会话状态包含标识会话特征的信息和握手协议的协商结果等，它包括如下元素：会话 ID、同等实体证书、压缩算法、密码规格（ciper-spec）、主密码（master secret）、是否可恢复标志（用于确定会话是否可用于初始化新连接的标志）。

11.4　TLS 握手协议

握手协议的功能是验证身份，协商密钥交换算法、压缩算法和加密算法，完成密钥交换以及生成密钥等。

11.4.1　基本消息描述

TSL 握手协议建立连接会话状态的密码参数，该过程在 TLS 记录协议之上进行，TLS 握手协议被封装在记录协议中。当 TLS 协议的客户机和服务器开始第一次通信时，首先需要协商协议版本，选择密码算法，进行相互认证(可选功能)，并使用公钥密码技术生成共享秘密。

这些消息交换能够实现如下操作：

(1) 客户机认证服务器；

(2) 允许客户机与服务器选择双方都支持的密码算法；

(3) 可选择的服务器认证客户机；

(4) 使用公钥加密技术生成共享密钥；

(5) 建立加密 TLS 连接。

TLS 握手协议报文头包括三个字段(见表 11-1)。

表 11-1　TLS 握手协议报文

报文类型	参　　数
hello_request	空
client_hello	版本、随机数、会话 ID、密文族、压缩方法
server_hello	版本、随机数、会话 ID、密文族、压缩方法
certificate	X.509v3 证书链
server_key_exchange	参数、签名
certificate_request	类型、授权
server_hello_done	空
certificate_verify	签名
client_key_exchange	参数、签名
finished	Hash 值

类型(1 字节)：该字段指明使用的 SSL 握手协议报文类型。SSL 握手协议报文包括 10 种类型。报文类型见表 11-1。

长度(3 字节)：以字节为单位的报文长度。

内容(大于等于 1 字节)：使用的报文的有关参数。

- hello 消息：服务器和客户机使用 hello 消息来交换安全相关的信息，如随机数、CipherSuite 等，包括 client_hello 消息、server_hello 消息和 hello_request 消息。
- server_certificate 消息：该消息表示服务器发送证书，在 server_hello 消息之后立即被发送。证书的类型必须与所选择的 CipherSuite 中的密码算法相匹配，一般情况

下是 X.509v3 格式的证书。

- server_key_exchange 消息：该消息仅在服务器发送的 server_certificate 消息中，没有包含足够的信息使客户可以交换 premaster_secret 的时候发送，为客户端协商 premaster_sectet 传递密码信息。
- certificate _request 消息：服务器发送该消息指示客户提供其证书。TLS 中证书类型包括 rsa_sign、dss_sign、rsa_fixed_dh、dss_fixed_dh 四种。
- server_hello_done 消息：服务器发送该消息指示 hello 阶段结束。
- client_certificate 消息：这是客户接收到 server_hello_done 消息之后能够发送的第一条消息，该消息只有在服务器要求证书的情况下才发送。如果客户，没有合适的证书，也可以发送不包含证书的 client_certificate 消息。
- client_key_exchange 消息：该消息由客户发送，一般使用 RSA 公钥加密的方式直接传输 premaster_secret，或者使用 Difffie-Hellman 方法使双方商定该密钥。
- certificate_verify 消息：该消息用来提供显式的客户证书校验。在 TLS 中 MD5 或 SHA-1 只对 handshake_messages 进行散列运算，如下所示：

```
CertificateVeryfy.signature.md5_hash MD5(handshake_messages)
Certificate.signature.sha_hash SHA(handshake_messages)
```

- finished 消息：该消息在 change_cipher_spec 消息之后发送，用来校验密钥交换和认证过程是否成功。

11.4.2 握手流程

TLS 握手协议的过程如下(如图 11-8 所示)。

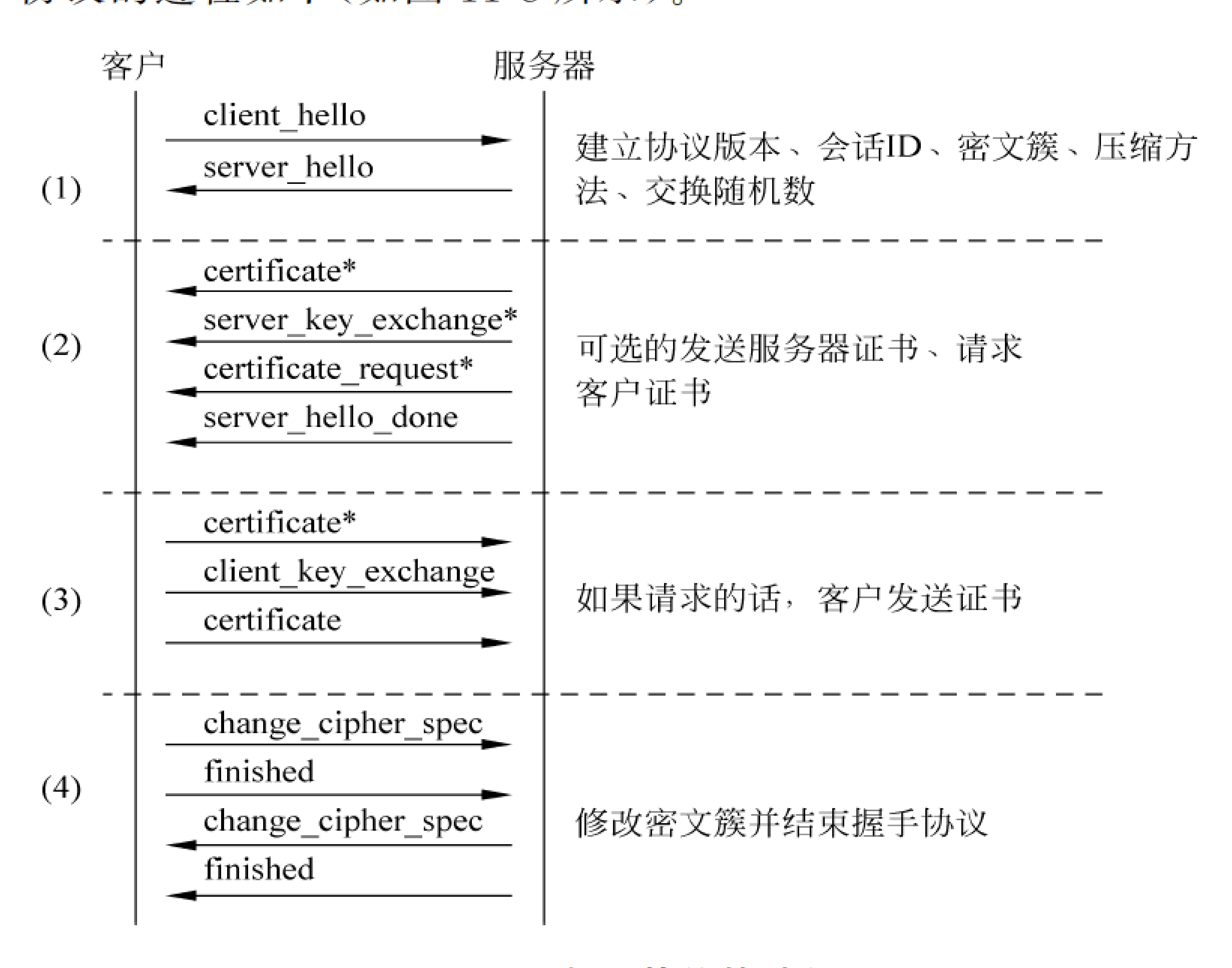


图 11-8 握手协议的过程

注：带 * 的传输是可选的，或者与站点相关的，并不总是发送的报文。

TLS 握手协议交换 hello 消息以协商密码算法，交换随机值并检查会话是否可重用；交换必要的密码参数，使客户和服务器能够协商预主秘密 premaster secret；交换证书和密码信息，使客户和服务器能够进行相互认证；使用交换的随机值和 premaster secret 生成主密钥 master secret；为记录协议提供安全参数；允许客户和服务器校验对方是否计算出了相同的安全参数，以及校验上述握手过程是否被攻击者窃听。

握手协议的过程作以下描述。

(1) 建立安全能力。客户机向服务器发送 client_hello 消息，服务器向客户机回应 server_hello 消息。client_hello 消息和 server_hello 消息建立如下的安全属性：协议版本、会话 ID、密文簇(CipherSuite)、压缩方法、同时生成并交换用于防止重放攻击的随机数。密文簇参数包括密钥交换方法(Diffie-Hellman 密钥交换算法、基于 RSA 的密钥交换和另一种实现在 Fortezza chip 上的密钥交换)、加密算法(DES、RC4、RC2、3DES 等)、MAC 算法(MD5 或 SHA-1)、加密类型(流或分组)等内容。

(2) 认证服务器和密钥交换。在 hello 消息之后，如果需要认证服务器的话，服务器将发送其证书。另外如果需要，服务器还要发送 server_key_exchange。对服务器进行认证后，服务器可以请求客户证书。然后服务器将发送 server_hello_done 消息，指示握手协议的 hello 消息阶段结束，服务器等待客户的反应。

(3) 认证客户和密钥交换。客户一旦收到服务器的 server_hello_done 消息，客户将检查服务器证书的合法性。如果服务器发送了 certificate_request 消息，客户必须发送客户证书，然后发送 client_key_exchange 消息，消息的内容依赖于 client_hello 与 server_hello 定义的密钥交换算法。如果客户发送了具有签名能力的证书，则需要发送 certificate_verify 消息显式地校验该证书。

(4) 结束。客户发送 change_cipher_spec 消息，并将挂起状态的 CipherSpec 复制到当前的 CipherSpec。这个消息使用的是改变密码格式协议。然后，客户在新的算法、对称密钥和 MAC 秘密之下立即发送 finished 消息。finished 消息验证密钥交换和鉴别过程是成功的。结束消息的内容如下：

PRF(master_secret, finished_label, MD5(handshake_messages) + SHA-1(handshake_messages))，这里的 finished_label 对于客户机来说是“client_finished”，对于服务器来说是“sever_finished”。

服务器响应客户的消息，发送自己的 change_cipher_spec 消息、finished 消息。握手结束，客户与服务器可以发送应用层数据了。

当客户从服务器端传送的证书中获得相关信息时，需要检查以下内容来完成对服务器的认证：时间是否在证书的合法期限内；签发证书的机关是否是客户端信任的；签发证书的公钥是否符合签发者的数字签名；证书中的服务器域名是否符合服务器自己真正的域名。服务器被验证成功后，客户继续进行握手过程。

同样地，服务器从客户传送的证书中获得相关信息认证客户的身份，需要检查的内容有：用户的公钥是否符合用户的数字签名；时间是否在证书的合法期限内；签发证书的机关是否是服务器信任的；用户的证书是否被列在服务器的 LDAP 里用户的信息中；得到验证的用户是否仍然有权限访问请求的服务器资源。

11.5 更改密码规格协议

该协议由单个消息 Change_Cipher_Spec 组成，消息中只包含一个值为 1 的单个字节。该消息的唯一作用就是使挂起状态的 ChipherSpec（密码规格）赋值为当前状态的 ChipherSpec，即将预生效的密码规范赋值为现行密码规范，更新用于当前连接的密码组。

客户和服务器都有各自独立的读状态(read state)和写状态(write state)。读状态中包含解压缩、解密、验证 MAC 的算法和解秘密钥等；写状态中包含压缩、加密、计算 MAC 的算法和加密密钥等。

在 SSL 中定义了两种状态：

① 未决状态(the pending state)包含了当前握手协议协商好的压缩、加密、计算 MAC 算法以及密钥等；

② 当前操作状态(the current operating state)包含了记录层正在实施的压缩、加密、计算 MAC 算法以及密钥等。

客户/服务器接收到 Change_Cipher_Spec 消息之后，立即把待定读状态中的内容复制至当前读状态；客户/服务器发送了 Change_Cipher_Spec 消息之后，立即把待定写状态中的内容复制至当前写状态。

11.6 警告协议

警告协议用来将 TLS 有关的警告传送给对方。警告协议的每个报文都由两个字节组成，第一字节指明级别(警告 Warning(1)和致命 Fatal(2))，第二字节指明特定警告的代码。Warning 类型的警告消息非常多，它的作用是通知另一方错误的出现。Fatal 类型的警告消息导致连接立即终止。在这种情况下，其他关于这个会话的连接可能还存在，但是会话标识必须失败，防止该会话再建立其他连接。第二个字节包含了指出特定警告的代码。

首先介绍致命警告。

- unexpected_message：接收了不合适的报文。
- bad_recod_mac：接收到错误的 MAC。
- decompression_failure：发送方不能产生可接收的安全参数组使选择可行。
- handshake_failure：解压缩函数输入不合适(例如，不能解压或超过最大允许长度的解压缩)。
- illegal_parameter：握手消息的某个超过值域或与其他的不相符。
- decryption_failed：密文解密的方式不正确，有可能是因为密文不是块长的偶数倍，或者填充值检测不正确。
- record_overflow：密文的记录长度大于 $2^{14}+2048$ 字节，或者密文解密后的长度大于 $2^{14}+2048$ 字节。
- unknown_ca：接收到有效证书链或部分链，但因为找不到 CA 证书或与已知的、可

信任的 CA 不匹配，不能接收这个证书。

- access_denied：接收到了一个有效的证书，但是当应用访问控制时，发送方决定不进行协商。
- decode_error：由于消息中的一些域超出了制定的范围，或消息的长度不正确。
- export_restriction：协商与检测到的出口限制不一致。
- protocol_version：客户端试图协商的版本能被识别，但不支持。
- insufficient_security：当握手失败是因为服务器端要求的安全强度高于客户端所支持的强度的时候，返回此消息，而不是 handshake_failure。
- internal_error：与另一端无关的内部错误，或是协议的修正使其无法继续。

下面介绍其余警告。

- decrypt_error：握手密码操作失败，包括不能验证签名、不能正确解密密钥交换、不能验证 finish 消息。
- user_canceled：握手的取消是与协议失败无关的原因。
- no_renegotiation：这条消息是在初始化握手以后由客户机或服务器发送的。此消息发送后，通常会导致重新协商。
- close_notify：通知接收方发送方在本连接中不会再发送任何消息。在关闭连接的写端前，每一方都需要发送一个 close_notify 警告。
- bad_certificate：接收到的证书已经被破坏(例如包含未经验证的签名)。
- unsupported_certificate：不支持接收的证书类型。
- certificate_revoked：证书已经被其签署者撤销。
- certificate_expired：证书已经过期。
- certificate_unknown：在实现证书时产生一些不确定的问题，使证书无法接收。

11.7　TLS 记录协议

TLS 记录协议不加解释地从高层接收非空的任意长度的数据块，并按照协议的规定对数据进行处理。

TLS 记录协议主要的操作如图 11-9 所示。

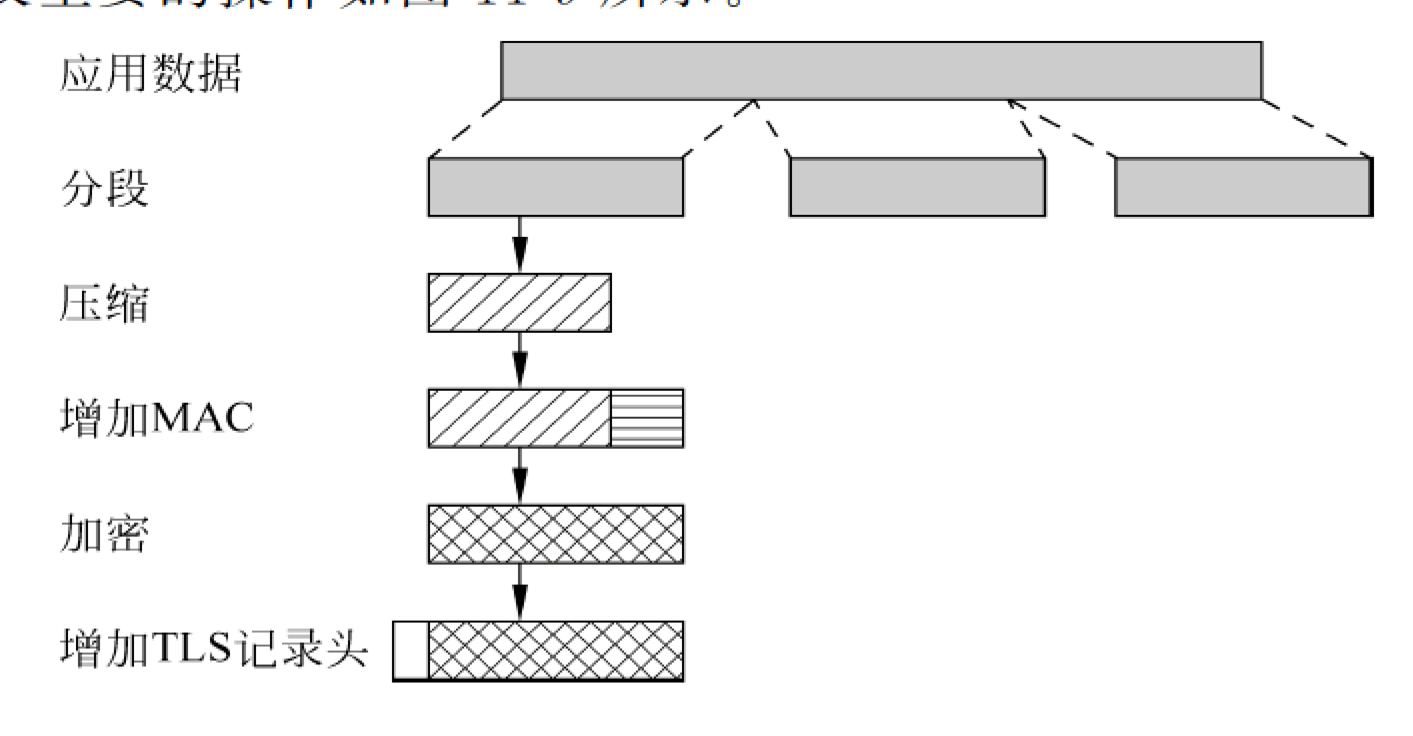


图 11-9　TLS 记录协议的操作

记录协议主要有以下功能。

1）分段

TLS 记录将每个上层应用数据分成小于 2^{14} 字节的 TLSPlaintext 记录。记录中包含类型、版本号、长度和数据等字段。

```
struct{
            uint8 major,minor;
          }ProtocolVersion;
enum{
            change_cipher_spec(20),alert(21),handshake(22),
            application_data(23),(255)
          }ContentType;
struct{
            ContentType type;                           //用于处理分段的高级协议
            ProtocolVersion version;                    //正在使用的协议版本
            uint16 length;                              //分段的长度
            opaque fragment[TLSPlaintext.length];       //明文分段[明文长度]
}TLSPlaintext;
```

2）压缩

压缩算法将 TLSPlaintext 结构变换成 TLSCompressed 结构。压缩必须是无损压缩，且不能使内容增加超过 1024 字节。TLS 定义了压缩信息的数据结构，也包括类型、版本、长度等信息。当然，TLS 也支持不压缩的空操作。

```
struct{
            ContentType type;                           //与 TLSPlaintext.type 相同
            ProtocolVersion version;                    //与 TLSPlaintext.version 相同
            uint16 length;                              //压缩分段的长度
            opaque fragment[TLSCompressed.length];      //压缩分段
}TLSCompressed;
```

3）计算 MAC

给压缩数据计算消息认证码，即 MAC。在 TLS 中采用了 RFC2104 定义的 HMAC 算法。

对于 TLS，MAC 计算包含如下字段：

```
HMAC_hash(MAC_write_secret, seq_num || TLSCompressed.type || TLSCompressed.version ||
TLSCompressed.length || TLSCompressed.fragment)
```

其中：

- MAC_write_secret：共享的保密密钥。
- seq_num：消息的序列号。
- TLSCompressed.type：用于处理分段的高级协议。
- TLSCompressed.version：正在使用的协议版本。
- TLSCompressed.length：压缩分段的长度。
- TLSCompressed.fragment：压缩分段（如果没有，就是明文分段）。

4）给加上 MAC 的压缩消息加密（加密采用对称密码，注意 MAC 总是在有效数据载荷

被加密之前被加入 TLS 记录中的）

加密的方式有两种：第一种是序列密码，第二种是分组密码。在分组密码中，为了使加密的数据大小是被加密块长的倍数，需要在 MAC 之后加一些填充字节。填充块之前有一个字节指示填充字节的长度。填充块的总量就是使加密的数据总量（元数据＋MAC＋填充块）是块密码长度倍数的最小字节数。

5）生成一个 TSL 记录报头

加密函数及消息认证码函数将 TLSCompressed 结构变换成 TLSCiphertext 结构。TLSCiphertext.fragment 的长度不应该超过 2048 字节。需要注意的是，应当在加密数据之前计算消息认证码 MAC。

```
struct{
          ContentType type;
          ProtocolVersion version;
          uint16 length;
          select(CipherSpec.cipher_type) {
              case stream: GenericStreamCipher;
              case block: GenericBlockCipher;
          }fragment;
}TLSCiphertext;
```

图 11-10 表示的是 TLS 记录协议的内容。

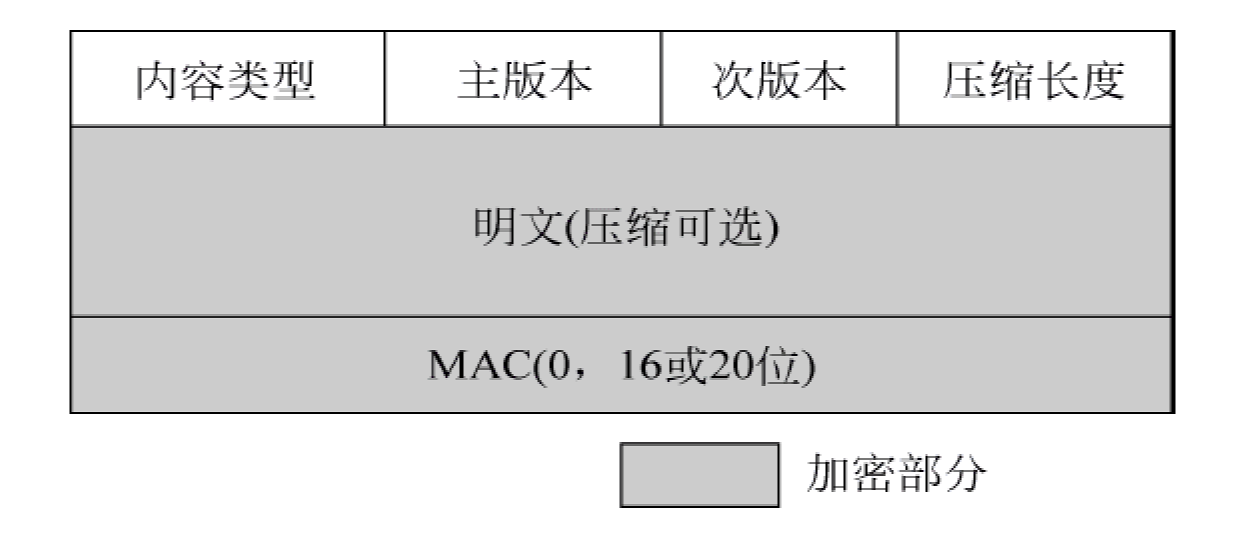


图 11-10　TLS 记录协议字段

内容类型（8 位）：封装的高层协议。已经定义的内容类型是握手协议、警告协议、改变密码格式协议和应用数据协议。

主要版本（8 位）及次要版本（8 位）：使用的 TLS 版本。对于 TLS 1.0，使用版本{3，1}。使用值{3，1}是有历史原因的，版本 TLS 1.0 是对 SSL 3.0 的微小的修改。

压缩长度：定义了原文分段的字节长度，最大是 $2^{14}+2048$。

11.8　TLS 协议中采用的加密和认证算法

1. 加密算法和会话密钥

共享主密码是通过安全密钥交换生成的临时 48 位组值。生成过程分为两步：第一步，交换 pre_master_secret；第二步，双方计算 master_secret。对于 pre_master_secret 交换，有两种可能性。

RSA：客户机生成48字节的pre_master_secret，用服务器的公共RSA密钥加密后，发送到服务器。服务器用私钥解密密码以恢复pre_master_secret。

Diffie-Hellman：客户机和服务器都生成Diffie-Hellman公钥，交换后双方都用Diffie-Hellman算法生成共享的pre_master_secret。

双方的master_secret计算如下：

```
master_secret = PRF(pre_master_secret, "master secret", ClientHello.radom + ServerHello.
radom)
```

虽然不同的密钥交换算法产生不同长度的pre_master_secret，但master_secret却始终是48字节。以上算法将一直执行，直到产生48字节的伪随机输出。

密钥块(包括会话加密密钥、MAC密钥、IV)的计算定义如下：

```
key_block = PRF(SecurityParameters.master_secret, "key expansion", SecurityParameters.server
_radom + SecurityParameters.client_radom)
```

这个计算结果也将一直执行，直到生成足够的输出。

2. 认证算法

认证算法采用X.509电子证书标准，通过RSA算法进行数字签名来实现。

1) 服务器的认证

服务器方的写密钥和客户方的读密钥、客户方的写密钥和服务器方的读密钥分别是一个私钥/公钥对。在握手协议的服务器身份阶段，服务器在发往客户机的Sever_Key_Exchange消息中包含了用户自己私钥加密的数字签名。具体方法是：先计算散列值hash(ClientHello.radom || SeverHello.radom || SeverParams)。在此散列值中，不但有Diffie-Hellman或RSA参数，而且包含了初始呼叫的两个现时值。这确保了对重放攻击和误传的防范。在DSS签名情况下，采用SHA-1散列算法；在RSA签名的情况下，采用MD5和SHA-1算法都能进行计算。计算之后的散列值用服务器的公钥进行签名。

2) 客户的认证

只有正确的客户方的密钥的内容才能被服务器方用其密钥正确地解开。当客户方收到服务器方发出的Request_Certificate消息时，客户首先使用MD5消息散列函数获得服务器消息的摘要，然后客户使用自己的私钥加密摘要形成数字签名，从而使自己的身份被服务器认证。

习 题 11

1. 在平时，您是否遇到过Web攻击，都是哪些类型的Web攻击？
2. CGI的工作原理是怎样的？
3. Web攻击主要有哪些？
4. 描述一下SSL协议访问网页的过程。
5. TLS握手协议的工作流程是怎样的？
6. TLS警告协议的作用是什么？
7. TLS记录协议主要完成哪些功能？

参 考 文 献

[1] 曹天杰,张永平,毕方明.计算机系统安全.2版.北京:高等教育出版社,2007.

[2] 方勇等.信息系统安全理论与技术.2版.北京:高等教育出版社,2008.

[3] 马建峰,郭渊博.计算机系统安全.西安:西安电子科技大学出版社,2005.

[4] 张红旗,王鲁.信息完全技术.北京:高等教育出版社,2008.

[5] Michael E Whitman, Herbert J Mattord. 信息安全原理. 2版. 齐立博译. 北京:清华大学出版社,2006.

[6] R Anderson. Security Engineering, Wiley, 2001.

[7] Lorrie Faith Cranor, Simson Garfinkel. Security and Usability. O'Reilly, August 2005.

[8] William Stallings. Cryptography and Network Security Principles and Practices (Fourth Edition). Prentice Hall, 2005.

[9] Charles P. Pfleeger-Pfleeger Consulting Group, Shari Lawrence Pfleeger-Rand Corporation. Security in Computing. Fourth Edition. Prentice Hall, 2006.

[10] Mark Stamp. Information Security Principles and Practice. Published by John Wiley & Sons, Inc., Hoboken, New Jersey. Published simultaneously in Canada, May 2005.

[11] Trent Jaeger. Operating System Security. Synthesis Lectures on Information Security, Privacy and Trust #1, 2008.

[12] Eric Cole, Ronald Krutz, James W Conley. Network Security Bible. Wiley Publishing, Inc. 2005.

[13] Thomas R Peltier, Justin Peltier, John Blackley. Information Security Fundamentals. CRC Press, LLC, Boca Raton London New York Washington, D. C., 2005.

[14] Abraham Silberschats, Peter Baer Galvin, Greg Gagne. 操作系统概念. 6版. Operating System Concepts. Sixth Edition. 郑扣根译.北京:高等教育出版社,2003.

[15] Joseph Migga Rizza. Computer Network Security. 2005 Springer Science+Business Media, Inc. USA.

21 世纪高等学校数字媒体专业规划教材

ISBN	书　　名	定价(元)
9787302224877	数字动画编导制作	29.50
9787302222651	数字图像处理技术	35.00
9787302218562	动态网页设计与制作	35.00
9787302222644	J2ME 手机游戏开发技术与实践	36.00
9787302217343	Flash 多媒体课件制作教程	29.50
9787302208037	Photoshop CS4 中文版上机必做练习	99.00
9787302210399	数字音视频资源的设计与制作	25.00
9787302201076	Flash 动画设计与制作	29.50
9787302174530	网页设计与制作	29.50
9787302185406	网页设计与制作实践教程	35.00
9787302180319	非线性编辑原理与技术	25.00
9787302168119	数字媒体技术导论	32.00
9787302155188	多媒体技术与应用	25.00

以上教材样书可以免费赠送给授课教师，如果需要，请发电子邮件与我们联系。

教学资源支持

敬爱的教师：

感谢您一直以来对清华版计算机教材的支持和爱护。为了配合本课程的教学需要，本教材配有配套的电子教案(素材)，有需求的教师可以与我们联系，我们将向使用本教材进行教学的教师免费赠送电子教案(素材)，希望有助于教学活动的开展。

相关信息请拨打电话 010-62776969 或发送电子邮件至 weijj@tup.tsinghua.edu.cn 咨询，也可以到清华大学出版社主页(http://www.tup.com.cn 或 http://www.tup.tsinghua.edu.cn)上查询和下载。

如果您在使用本教材的过程中遇到了什么问题，或者有相关教材出版计划，也请您发邮件或来信告诉我们，以便我们更好地为您服务。

地址：北京市海淀区双清路学研大厦 A 座 708　　计算机与信息分社魏江江　收
邮编：100084　　电子邮件：weijj@tup.tsinghua.edu.cn
电话：010-62770175-4604　　邮购电话：010-62786544

《网页设计与制作》目录

ISBN 978-7-302-17453-0　　蔡立燕　梁　芳　主编

图书简介：

Dreamweaver 8、Fireworks 8 和 Flash 8 是 Macromedia 公司为网页制作人员研制的新一代网页设计软件，被称为网页制作"三剑客"。它们在专业网页制作、网页图形处理、矢量动画以及 Web 编程等领域中占有十分重要的地位。



本书共 11 章，从基础网络知识出发，从网站规划开始，重点介绍了使用"网页三剑客"制作网页的方法。内容包括了网页设计基础、HTML 语言基础、使用 Dreamweaver 8 管理站点和制作网页、使用 Fireworks 8 处理网页图像、使用 Flash 8 制作动画、动态交互式网页的制作，以及网站制作的综合应用。

本书遵循循序渐进的原则，通过实例结合基础知识讲解的方法介绍了网页设计与制作的基础知识和基本操作技能，在每章的后面都提供了配套的习题。

为了方便教学和读者上机操作练习，作者还编写了《网页设计与制作实践教程》一书，作为与本书配套的实验教材。另外，还有与本书配套的电子课件，供教师教学参考。

本书适合应用型本科院校、高职高专院校作为教材使用，也可作为自学网页制作技术的教材使用。

目　录：